**Seed Germination Monitoring System**

From: Christian Katsabas & Derek McCreery  
Discipline: Computer Engineering Technology  
Date: 4/21/2016

# Declaration of Joint Authorship

We, Christian Katsabas and Derek McCreery, confirm that this work submitted for assessment is the joint work of ourselves, and is expressed in our own words. Any uses made within of other works of any other author, in any form (ideas, equations, figures, previous technologies, tables, programs, texts) are properly acknowledged at the point of use. A list of the references used is included. Christian Katsabas has handled the Web Interface and Database, and Derek McCreery has handled the software, hardware, and mobile application aspects of this project.

# Approved Proposal

## Executive Summary

As a student in the Computer Engineering Technology program, I will be integrating the knowledge and skills I have learned from our program into this Internet of Things themed capstone project. This proposal requests the approval to build the hardware portion that will connect to a LAMP server via ZeroTier as well as to a mobile device application. The internet connected hardware will include a custom PCB with sensors and actuators for the measuring of humidity, moisture, light, temperature, and water level. The MySQL database will store the data generated from the sensors. The mobile device functionality will include the ability to see the current data and the data from the past. I will be collaborating with Valeria Wuschnakowski, greenhouse technician at Humber. In the winter semester I plan to form a group with the following student, who is also building similar hardware this term Christian Katsabas. The hardware will be completed in CENG 317 Hardware Production Techniques independently and the application will be completed in CENG 319 Software Project. These will be integrated together in the subsequent term in CENG 355 Computer Systems Project as a member of a 2 student group.

## Background

The problem solved by project is that that there is no current system in the greenhouse at Humber to measure the humidity, moisture, light, temperature, and water level which would be needed to ensure that the plants grow. My project will help the technicians in the greenhouse determine under which circumstances the germination process best happens under. By having the monitoring system in place, it will be easy to determine not only if the plants need to be taken care of right now, but also identify at what times of the day the plants needs extra attention. For example, the plants may need to be watered more during the mid-day hours as opposed to the afternoon or night. The germination process is when the plant is growing from a seed.

I have searched for prior art via Humber’s IEEE subscription selecting “My Subscribed Content” and have found and read which provides insight into similar efforts.

## Concluding remarks

This proposal presents a plan for providing a solution for the green house at Humber College. This is an opportunity to integrate the knowledge and skills developed in our program to create a collaborative capstone project demonstrating my ability to learn how to support projects. I request approval of this project.

# Abstract

There is a large need for small-scale agricultural operations to constantly monitor the environment around newly growing plants. To remedy this, a system should be created to actively monitor the environment during the germination process of a plant in a way that allows a user to remain up to date on the environment without having to be present. This system monitor ambient temperature, relative humidity, light levels towards the plant, and the moisture of the soil, and allow the plant to be watered should the soil moisture drop too low. These sensors will store the data temporarily on the Development Platform before it is sent to an offsite database. This database will hold all of the sensor data, and make it available to both an Android application, and a website. The application will display the most recent data entries, as well as a limited history of data entries from the database, including graphs for ease of viewing. The website will display the most recent batch of entries, and provide the full history for each individual sensor’s data, as well as a graph for ease of viewing trends. This system has the potential to make small-scale growing operations easier to monitor, allowing users to check their plants’ conditions from anywhere at any time, provided they have internet connectivity.

Table of Contents

[Declaration of Joint Authorship iii](#_Toc449009667)

[Approved Proposal v](#_Toc449009668)

[Executive Summary v](#_Toc449009669)

[Background v](#_Toc449009670)

[Concluding remarks vi](#_Toc449009671)

[Abstract vii](#_Toc449009672)

[Illustration List xiii](#_Toc449009673)

[1. Introduction 1](#_Toc449009674)

[2. Project Description 3](#_Toc449009675)

[2.1 Problem 3](#_Toc449009676)

[2.2 Rationale Behind Project 3](#_Toc449009677)

[2.3 Project Scope 3](#_Toc449009678)

[2.4 Software Requirement Specifications 4](#_Toc449009679)

[2.4.1 Database 4](#_Toc449009680)

[2.4.2 Mobile Application 5](#_Toc449009681)

[2.4.3 Web Interface 5](#_Toc449009682)

[2.4.4 Networked Platform Communication Software 6](#_Toc449009683)

[2.4.5 Microcontroller Firmware 7](#_Toc449009684)

[2.5 Project Overview 7](#_Toc449009685)

[2.5.1 Bill of Materials 7](#_Toc449009686)

[2.5.2 Time Commitment 8](#_Toc449009687)

[2.5.3 Mechanical Assembly 8](#_Toc449009688)

[2.5.4 PCB and Soldering 12](#_Toc449009689)

[2.5.5 Power Up 12](#_Toc449009690)

[2.5.6 Unit Testing 13](#_Toc449009691)

[2.5.7 Production Testing 14](#_Toc449009692)

[2.6 Problems Encountered 15](#_Toc449009693)

[2.6.1 Growth Medium Moisture 15](#_Toc449009694)

[2.6.2 Leaks in Casing 15](#_Toc449009695)

[2.6.3 Limited Space for Electronics 15](#_Toc449009696)

[2.6.4 Data Communication 15](#_Toc449009697)

[2.6.5 Website Graphing 16](#_Toc449009698)

[2.7 Approaches 17](#_Toc449009699)

[2.7.1 Growth Medium Moisture 17](#_Toc449009700)

[2.7.2 Leaks in Casing 17](#_Toc449009701)

[2.7.3 Limited Space for Electronics 17](#_Toc449009702)

[2.7.4 Local Data Communication 18](#_Toc449009703)

[2.7.5 Website Graphing 18](#_Toc449009704)

[2.8 Walkthrough of System 18](#_Toc449009705)

[2.8.1 Microcontroller 18](#_Toc449009706)

[2.8.2 Microprocessor 18](#_Toc449009707)

[2.8.3 Server 19](#_Toc449009708)

[2.8.4 Phone Application 19](#_Toc449009709)

[2.8.5 Website 19](#_Toc449009710)

[3. Progress Reports 20](#_Toc449009711)

[3.1 Report 1 20](#_Toc449009712)

[3.2 Report 2 20](#_Toc449009713)

[3.3 Report 3 22](#_Toc449009714)

[3.4 Report 4 23](#_Toc449009715)

[3.5 Report 5 24](#_Toc449009716)

[3.6 Report 6 25](#_Toc449009717)

[4. Conclusions 27](#_Toc449009718)

[5. Recommendations 28](#_Toc449009719)

[6. Technical References 30](#_Toc449009720)

[7. Appendicies 32](#_Toc449009721)

[7.1 Microcontroller Firmware 32](#_Toc449009722)

[7.2 Microprocessor Communication Script 38](#_Toc449009723)

[7.3 Microprocessor Database Communication Program 40](#_Toc449009724)

[7.4 Database Input Script 43](#_Toc449009725)

[7.5 Database Retrieval Script for Phone Application 45](#_Toc449009726)

[7.6 Website Code 48](#_Toc449009727)

[7.6.1 Main Page 48](#_Toc449009728)

[7.6.2 Station Page 51](#_Toc449009729)

[7.6.3 Humidity Page 54](#_Toc449009730)

[7.6.4 Light Page 60](#_Toc449009731)

[7.6.5 Temperature Page 66](#_Toc449009732)

[7.6.6 Moisture Page 72](#_Toc449009733)

[7.7 Android Phone Application 78](#_Toc449009734)

[7.7.1 Main Menu 78](#_Toc449009735)

[7.7.2 Current Data 92](#_Toc449009736)

[7.7.3 Humidity History 96](#_Toc449009737)

[7.7.4 Humidity Graph 100](#_Toc449009738)

[7.7.5 Light History 104](#_Toc449009739)

[7.7.6 Light Graph 108](#_Toc449009740)

[7.7.7 Soil Moisture History 112](#_Toc449009741)

[7.7.8 Soil Moisture Graph 116](#_Toc449009742)

[7.7.9 Temperature History 120](#_Toc449009743)

[7.7.10 Temperature Graph 124](#_Toc449009744)

[7.7.11 Style 128](#_Toc449009745)

[7.7.12 Text 129](#_Toc449009746)

[7.7.13 Vertical Main Menu Layout 130](#_Toc449009747)

[7.7.14 Horizontal Main Menu Layout 134](#_Toc449009748)

[7.7.15 Vertical Current Data Layout 138](#_Toc449009749)

[7.7.16 Horizontal Current Data Layout 143](#_Toc449009750)

[7.7.17 Humidity History Layout 149](#_Toc449009751)

[7.7.18 Light History Layout 151](#_Toc449009752)

[7.7.19 Moisture History Layout 153](#_Toc449009753)

[7.7.20 Temperature History Layout 155](#_Toc449009754)

[7.7.21 Humidity Graph Layout 157](#_Toc449009755)

[7.7.22 Light Graph Layout 158](#_Toc449009756)

[7.7.23 Moisture Graph Layout 159](#_Toc449009757)

[7.7.24 Temperature Graph Layout 160](#_Toc449009758)

# Illustration List

2.4a Layout design of the website. p.3

2.5.3a Circuit layout for supplying power to pump. p.5

2.5.3b Circuit layout for sensor connections to microcontroller board. p.6

2.5.3c Finished prototype in acrylic case. p.7

2.5.7a Components in acrylic case before PCB and dividers. p.9

# 1. Introduction

The task of monitoring a plant’s growth can be tedious, and requires a large number of man-hours to maintain the environment and monitor the plant’s growth. Small scale growing operations, while perhaps more able to control their environment, may have trouble in finding the time to constantly monitor their systems, or even being able to afford the complex and often expensive systems used to monitor and control environmental factors. To this end, we have developed an integrated solution with a relatively low cost per plant that will allow for close monitoring of each plant’s unique environment. In addition, the system will water the plants as needed, further cutting time for plant maintenance.

Our system will store the data in an off-site database, which will then be accessible via both a website interface, and Smartphone application, providing on-the-go monitoring of the plant. This will provide growers with a peace of mind they would otherwise be unable to have through an in-house only monitoring system.

Despite possible redundancy of our system due to products which both monitor and control environmental factors [1][2], we nonetheless feel that our system will be valuable to those that cannot afford such complex, and often costly products. It will provide a lightweight, small-scale monitoring system, which will be valuable not only to small-scale growth operations, but to even amateur gardeners in their own homes.

Due to dealing with water, our system needed to be able to isolate our electronics from potential sources of water damage, which we solved by creating waterproofed compartments within the box used for the system.

In accordance with GPLv2, we will make all source code for our firmware and Android Application available, due to use of the WiringPi, GraphView, and Rickshaw libraries [3][4][5].

# 2. Project Description

## 2.1 Problem

Monitoring multiple, potentially hundreds of plants, especially in a small scale arboreal setup, can cost unnecessarily large amounts of man-hours. Most currently existing systems to automatically monitor plant and seed growth are either cumbersome to set up, difficult for a non-professional to use, or expensive to purchase and/or maintain.

## 2.2 Rationale Behind Project

In order to solve this problem, a small-scale, inexpensive, automatic plant growth monitoring system was devised. This system will be capable of gathering data comparable in detail and accuracy to larger, more expensive systems that have been deployed in past.

## 2.3 Project Scope

To meet these requirements, this system has been developed with a certain set of criteria that had to be met in order to solve the problem. The system must be able to measure, with comparable accuracy to similar marketed systems, ambient temperature, relative humidity, intensity of light, and moisture content of the growth medium. Should the sensor measuring moisture content detect a deficiency of moisture in the growth medium, the system must be able to supply water to the medium, in order to avoid both the medium drying out and causing damage to the plants being grown. In order to monitor trends in data for both record keeping and informative purposes, the system must be able to transmit data to a database, which must contain time and date stamps in addition to sensor data. To facilitate the online interface and smart phone application interface, these two elements must be able to retrieve information from the database.

Due to the intention of producing an inexpensive and easy-to-use system, certain features have been deemed currently infeasible for the scope of this system. The system does not need to be able to measure the water remaining in its reservoir. The system does not need to be able to recycle water from the growth medium back into the reservoir. The system does not need to be able to control the ambient temperature, the relative humidity, or the light intensity of the environment around the growth medium. The system does not need to be able to present a real-time view of the growing plants, or images of the plants. The system does not need to be able to provide an option for the user of the system to manually water the plants in the growth medium.

## 2.4 Software Requirement Specifications

### 2.4.1 Database

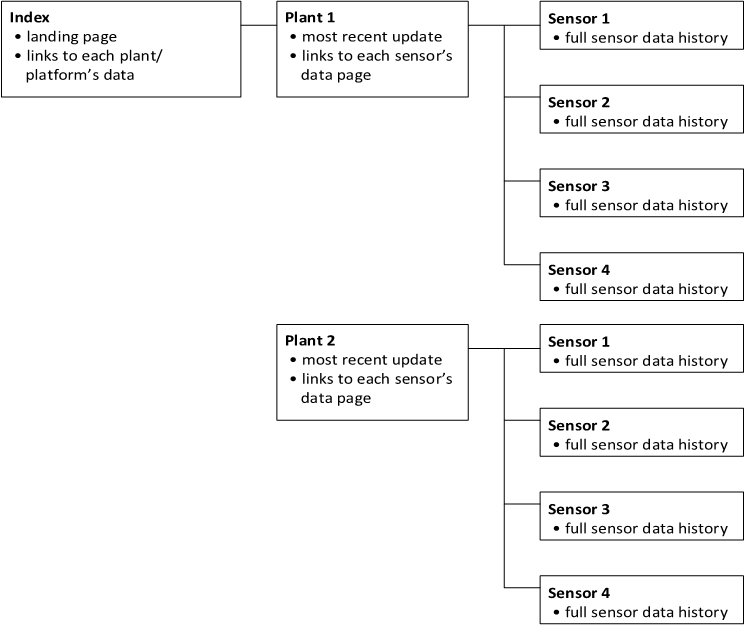
There will be a MySQL Database located on a co-located development platform. This database will be connected to a front-end administrated by phpMyAdmin. This database will hold the formatted data from the attached development platform, as well as a date and time stamp for each entry. The database will contain a single table, and the fields in the table will be: which plant/unit the data is from, the date and timestamp, the temperature, the ambient humidity, the soil moisture level, and the light level. (Developed by Christian Katsabas)

### 2.4.2 Mobile Application

There will be a mobile application (currently only available on Android platforms) which will take the formatted data from the database, as well as the date and time stamp, and display them in an easy-to-read manner, separated by sensor, including a graph view of data. The user will be able to customize the range of history from the database entries that will be stored by the app. There will be a view in the application which will display all four sensor information the most recent entry from the database, as well as unique views for the four main environmental factors: temperature, humidity, soil moisture, and light levels. The application will call for an update from the database every time it is run, and will be able to manually request an update from the database on user request. (Developed by Derek McCreery)

### 2.4.3 Web Interface

There will be a web interface which will take the formatted data from the database, as well as the date and time stamp, and display them in an easy-to-read manner, separated by sensor. Each plant will have its own page with an overview of the most recent few data updates, with each sensor having its own page under the plant’s top-level page. The web interface will have the full database’s contents available to review. The web interface will update from the database on page load. (Developed by Christian Katsabas)

  
Image 2.4a: Website layout diagram

### 2.4.4 Networked Platform Communication Software

There will be software running on the attached development platform, which will take in the data from the microcontroller and write it to a file. A separate program will then read from the aforementioned file and send it to the database to be then read from the mobile application and the website.(Developed by Derek McCreery)

### 2.4.5 Microcontroller Firmware

There will be software running on the microcontroller, which will read raw data from the sensors, format the data into standard units, and send the data to the attached development platform. The firmware will measure ambient temperature in °C, ambient humidity in percentage, light levels in lux, and the moisture of the growth medium in percentage. After reading the moisture content of the growth medium, the microcontroller determines if it needs to turn on the pump to water the growth medium and also for how long.(Developed by Derek McCreery)

## 2.5 Project Overview

### 2.5.1 Bill of Materials

[Arduino Uno - $28.99](http://www.amazon.ca/Arduino-A000066-Uno-R3-Microcontroller/dp/B008GRTSV6/ref=sr_1_1?ie=UTF8&qid=1449726852&sr=8-1&keywords=arduino+uno)  
[Raspberry Pi starter kit - $89.95](http://www.amazon.ca/gp/product/B00MV6TAJI?psc=1&redirect=true&ref_=oh_aui_detailpage_o05_s00)   
[USB cable (Pi to Arduino) - $3.95](https://www.adafruit.com/products/62)   
[Peristaltic liquid pump - $24.95](https://www.adafruit.com/products/1150?&main_page=product_info&products_id=1150)   
[Power adapter (Arduino) - $6.95](https://www.adafruit.com/products/63)   
[1N4004 Diode - $1.50](https://www.adafruit.com/products/755?&main_page=product_info&products_id=755)   
[TMP36 – Analog Temperature sensor - $1.50](https://www.adafruit.com/products/165)   
 [SparkFun Humidity Sensor Breakout - HIH-4030 - $16.95](https://www.sparkfun.com/products/9569)   
[GA1A12S202 Log-scale Analog Light sensor - $3.95](https://www.adafruit.com/product/1384)   
[Soil moisture sensor - $1.93 (I recommend ordering from Phantom YoYo as it is about half the cost!)](http://www.amazon.ca/gp/product/B00AFCNR3U?psc=1&redirect=true&ref_=oh_aui_detailpage_o02_s00)   
[MTP3055VLMOSFET - $1.65](http://www.digikey.ca/product-detail/en/MTP3055VL/MTP3055VLFS-ND/1055090)   
[470kΩ Resistor - $4.44 (You only need a single resistor between 100kΩ and 1MΩ)](http://www.amazon.com/E-Projects-470k-Resistors-Watt-Pieces/dp/B00IYU99Q4/ref=sr_1_9?s=industrial&ie=UTF8&qid=1449730299&sr=1-9&keywords=470K+Ohm+Resistors)   
[Hook-up wire - $2.50](https://www.adafruit.com/products/289)

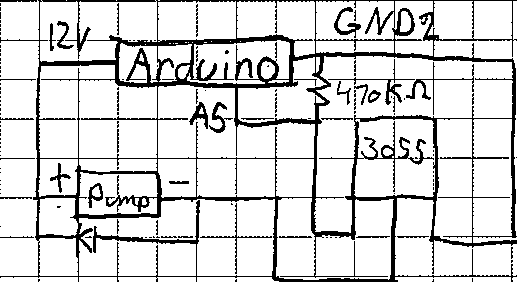
Prices may differ based on currency, time, and supplier.

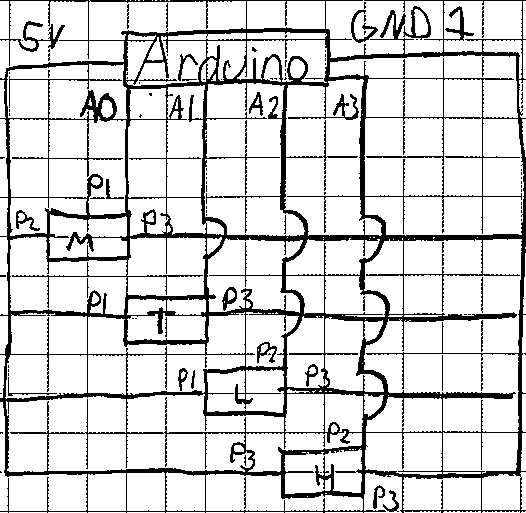
### 2.5.2 Time Commitment

The time it would take to reproduce the project is about six hours is using the code provided and if you already have the materials, longer if you want to create the code by yourself. If using the code provided, some slight modifications need to be made to write to your own server. This approximation is made on someone who understands the basics of coding and soldering. This time can vary greatly depending on a person’s familiarity working on projects similar in scope to this one.

### 2.5.3 Mechanical Assembly

The following is a guide on how the circuit is assembled should you choose not to use a PCB or wish to create a prototype of the circuit on a breadboard. This section can also be referred to when checking that proper connection were made when making revisions to the PCB.

The pump:   
Connect the Vin pin on the Arduino to the positive pin on the pump, next connect the negative pin on the pump to the drain pin of the MOSFET. Connect a diode across the two pump pins so that the negative side of the diode is towards the positive pin on the pump. Connect the gate pin on the MOSFET to the Arduino analog pin 5. Put a 470kΩ from the gate pin to ground. Connect the source pin to ground. Below is a circuit diagram to illustrate how to set the circuit up.  
  
Image 2.5.3a: Circuit layout for supplying power to pump.  
  
The sensors:   
Connect all of the sensor's 5V pins to the fixed 5V pin of the Arduino as well as the Vin of the light sensor. Next, connect them all to a common ground (I chose to use separate grounds for the pump and the sensors, to make use of the 2 ground pins on the Arduino board.). Next, connect the data pins to their respective pins as shown in the Arduino .ino file included in the project folder. The picture below is a visual representation of these instructions.

  
Image 2.5.3b: Circuit layout for sensor connections to microcontroller board.  
**IMPORTANT NOTE: Because the light sensor does not return a value greater than 3V to the Arduino, connect the 3.3V pin to the AREF pin on the opposite side of the board. The code will use this connection to get a more accurate reading of the light level.**   
  
The box:   
Included in the Project Files folder are the files to have a laser cutter cut out the dividers for the box shown. The included files contain the files for the box itself as well as the dividers that are to be places inside the box to create the separate compartments. The long piece is a middle divider between the soil and the pump and the water reservoir. The two identical pieces are to make small compartments for the pump and reservoir to make sure that no water is spilled onto the electronics. The last piece is a platform that I made to allow me to stack the Arduino on top of the Raspberry Pi to allow them to both fit in the compartment and still have enough room for the wires that connect the two platforms together as well as power and an hdmi cable to allow the Raspberry Pi to display video output to a monitor. I used Gorilla Glue to hold together the pieces of acrylic and to waterproof the compartments to ensure that no water leaked onto the Pi. Any sort of sealant would work; just ensure that it is waterproof. Be sure to know if the sealant chosen expands as it dries as this could push apart the connect pieces and no longer create a waterproof seal.  
  
Image 2.5.3c: Finished prototype in acrylic case.

### 2.5.4 PCB and Soldering

The soil moisture sensor will be connected to the board by hook-up wires, so ensure that the wires used are long enough to reach from the sensor, to the soil. The temperature, humidity, and light sensors should be mounted on the top side of the board. The light sensor will be a little tricky to solder because you need to make a secure connection from the 5V line to the top side of the board as the humidity sensor receives 5V on the top side. Ensure when soldering in the MTP3055VL MOSFET that the side with writing on it is facing the resistor, and that the cathode of the diode is connected to the positive pin of the pump.

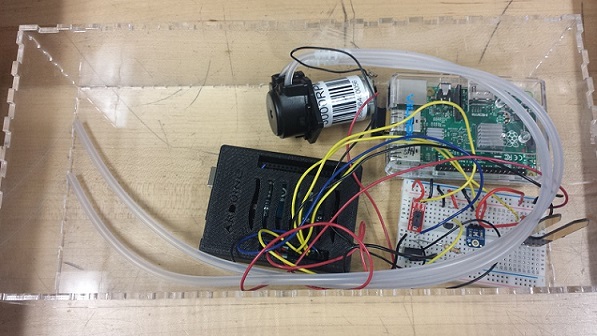
### 2.5.5 Power Up

Ensure that there are no shorts on the PCB itself before hooking up power to the Arduino. This can be done by using an Ohmmeter and checking the resistance between the traces and ensuring that there are no shorts (There will be a very small resistance, almost 0Ω, if there is a short). After this is done, turn on the Pi and open a terminal. Navigate to the folder containing the Python script and type ***python readinputs.py*** but do not run the command yet. Plug in the Arduino's power supply cable FIRST, then connect to the Pi through the USB port on both devices. Once they are connected, run the script and you should be seeing the data on the screen. If the data seems off, the script may not be synced up with the display. To solve this, run the script again after the data stops being transmitted and the output should be synced up.   
  
After booting up the Raspberry Pi for the first time and doing the initial set-up, type the command ***sudo apt-get update*** then when it is finished, run ***sudo apt-get dist-upgrade***. This will ensure that your platform is up to date and should be working as intended. Next you will need to download the Arduino IDE to upload your code to the Arduino Platform. Run the command ***sudo apt-get install arduino***. After setting up your Pi with all the programs you need, type the command ***startx*** to get to a more user-friendly GUI to help with file management and to use the Arduino IDE. Open the .ino file in from the Project Files folder with the Arduino IDE and upload the file from the Pi to the Arduino. Make sure the USB cable is connecting the two devices. Next, run the python script by typing in the command line ***python readinputs.py*** and this will display the data on the command line as it is sent from the Arduino. If the data is out of sync (it should be waiting on the soil moisture to be sent), restart the python script, this should have the program sync back up.   
  
If you want the data to be sent to a database, compile the java program by typing ***javac writedata.java***. Next, modify the files in the Server Programs subfolder and use the command ***python readinputsSendToData.py*** to send the data to the ErgoWrite.php script that is on the server. This will write the data into the database. ErgoData.php is used by the phone application to read from the database.

### 2.5.6 Unit Testing

The temperature sensor should be returning the correct value of the current temperature in degrees Celsius. The humidity and soil moisture should be returning a value between 0 and 100, which is the percent value. The light sensor should be returning the current light level in lux. The light value if you are in a moderately lit room should be around 80lux, the soil moisture should read 0% if it is not placed in soil. Should any of the values be returning incorrectly, ensure that the sensor is placed in the mounting correctly, and that the solder is correctly applied.

### 2.5.7 Production Testing

Ensure that the pump is turning on once every minute for 5 seconds and that the values returned form then sensors are correct. Should anything not be returning the correct value, or the pump is not turning on (When not in moist soil), follow the unit testing guide above to help troubleshoot your problem.   
Image 2.5.7a: Components in acrylic case before PCB and dividers.

## 2.6 Problems Encountered

### 2.6.1 Growth Medium Moisture

It proved difficult to directly measure the moisture of a growth medium, particularly in a large area. Without solving this issue, it would be difficult to ensure the growth medium stayed adequately moist for the growing plants.

### 2.6.2 Leaks in Casing

As the acrylic case is not originally watertight, there is the potential for water to leak through the dividers, as well as out of the case itself, particularly in the segment containing the growth medium. This can cause damage to any surface it rests on, as well as potentially causing damage to the electronic components of the system, should water leak through to that segment.

### 2.6.3 Limited Space for Electronics

As this system was required to take up no more than a certain volume of space, it was difficult to find a solution that would allow the electronics to remain relatively organized while still maintaining connections.

### 2.6.4 Data Communication

The system is made of two platforms that are required to communicate with one another to gather and transfer data to a server. Because of this, there were many problems with receiving the data in a way that it could then be easily sent to the database as the database also was not able to be connected through the project’s original plan.

### 2.6.5 Website Graphing

Displaying the data for users to look at and be able to quickly and easily understand is a goal of the website’s design. Graph’s were introduced as a solution to achieve this goal but there were difficulty in implementing them in a way that allowed them to display the gathered data completely dynamically.

## 2.7 Approaches

### 2.7.1 Growth Medium Moisture

To measure the moisture of the growth medium, the easiest solution that was found was to utilize a local measurement of the natural electrical resistance of the medium, using a dual-pronged probe with a reference resistor. This solution works on the principle that the moister a growth medium is, the less electrical resistance it has, allowing a voltage divider style solution to work effectively.

### 2.7.2 Leaks in Casing

To address the issue of the case leaking, this system used a strong adhesive, silicone-based epoxy to seal the gaps between each segment, and along the inner corners of the case. This effectively made the soil and reservoir compartments watertight at the bottom and sides.

### 2.7.3 Limited Space for Electronics

To address the size constraints presented by the case, the development platform was allowed to rest on the bottom of the case in its segment, whereas the microcontroller rested on an acrylic shelf directly above the development platform. This allows for shorter wires being potentially used for connections between the two, as well as allowing more room for additional peripheral wiring, such as audio/video connectors from the development platform for maintenance and debugging, additional potential room for the motor to be housed more securely in the wall of the divider, and for the sensor PCB to be laid on top of the microcontroller directly.

### 2.7.4 Local Data Communication

To solve this issue, a Python script was created to read the incoming data then write it into a file, then to call a Java program. This Java program then read the data from the file, and passed it on to a PHP script that was hosted on the server with the database which then was able to place the data into the database.

### 2.7.5 Website Graphing

To solve this issue we decided to have the graphs on the website display the same amount of data the phone does to allow us to hard-code the amount of data displayed. This allowed us to have the size of the graph a constant, while having the data it displayed dynamic.

## 2.8 Walkthrough of System

### 2.8.1 Microcontroller

The four sensors are attached to the PCB which is mounted onto the microcontroller. The microcontroller then gathers the readings from the four sensors every hour and runs the raw data through equations to determine the correct value of the reading in the appropriate units. This data is then sent to the microprocessor via serial communication using a USB connection.

### 2.8.2 Microprocessor

The microprocessor has a Python script that waits for data to be received through the serial connection. Once the data has been received, the script displayed the data to the terminal window and writes the data to a file. This file is overwritten every time to ensure that the older data is not sent to the database a second time. Once the file has been written to, the script then calls a Java program. This program then reads from the file and formats the data into a URL that is used to send the data to a PHP script on the server that is being used.

### 2.8.3 Server

The server receives the connection sent by the microprocessor’s Java program and runs a PHP script that reads the data sent to it though the URL. This data is then formatted into a SQL query and then inserted into the database. This returns a success message to the Java program on the microprocessor to give the user some feedback about the status of the data.

### 2.8.4 Phone Application

The phone application calls a PHP script on the server that returns the last 24 hours of entries in an array which is then parsed through and added to 4 local arrays within the app. These arrays are passed through each page of the app to display the data to the user through either a list or a graph. The Current Status screen displays only the most current entry of the database to the user.

### 2.8.5 Website

The website works similar to the phone application where it takes in the data from the database and displays it to the user through both a list and a graph. The difference between the two is that the website allows the user to see all of the data for each of the sensors since it has started writing to the database and not just the last twenty four readings. Refer to image 2.4a for a diagram of the website’s layout.

# 3. Progress Reports

## 3.1 Report 1

|  |  |
| --- | --- |
| **Derek McCreery** <derekmccreery@gmail.com> | Thu, Jan 21, 2016 at 2:03 PM |
| To: Kristian Medri <Kristian.Medri@humber.ca>  Cc: Christian Katsabas <n01001369@humbermail.ca> | |
| |  | | --- | | Dear Kristian,  This is our CENG 355 status update for the Seed Germination project.  Currently, we have planned out the software requirements for the project.  Sincerely,  Derek McCreery | | |

## 3.2 Report 2

|  |  |
| --- | --- |
| **Christian Katsabas** <ckatsabas@gmail.com> | Thu, Feb 4, 2016 at 3:08 PM |
| To: Kristian Medri <kristian.medri@humber.ca>  Cc: Derek McCreery <derekmccreery@gmail.com> | |
| |  | | --- | | Dear Kristian,  This is our CENG 355 status update for the Seed Germination project. As of this week, we have completed the SRS and it has been submitted. Currently, our firmware is in beta stage, and will be likely finalized by the end of reading week. We will be getting a re-print of our circuit board, and will be getting the box modified to fit our new design concept.  The mobile application has been basically completed, will need to be updated to connect to the database, however otherwise is complete. We may consider polishing the interface, however that is not currently a priority.  Our web interface is currently in the conceptual stage, will be implemented in rough with dummy data by the end of reading week, possibly connected to the database, pending database colocation.  The database has been completed on the coclocated development platforms we have been given for use at humber, however we plan to host the database off-site if possible, to ensure a higher connectivity consistency.  The introduction was mostly lifted from a combination of Derek and Christian's project proposals, and has been updated in the skeleton document.  Sincerely,  Christian Katsabas | | |

## 3.3 Report 3

|  |  |
| --- | --- |
| **Derek McCreery** <derekmccreery@gmail.com> | Thu, Feb 11, 2016 at 1:52 PM |
| To: Kristian Medri <Kristian.Medri@humber.ca>  Cc: ckatsabas@gmail.com | |
| |  | | --- | | This is the Seed Germinations project's status update for the week on February 11, 2016.  The firmware is still in the beta stage and is still on track to being completed by the end of the reading week.  The reprinted PCB has been made and the sensors have been remounted but still requires basic testing to ensure they are still working correctly.  The mobile application is completed but has yet to be modified to receive the data from the database.  Our web interface is currently in the conceptual stage, will be implemented in rough with test data by the end of reading week, possibly connected to the database, pending database colocation.  The database has been completed on the collocated development platforms we have been given for use at Humber, however we plan to host the database off-site if possible, to ensure a higher connectivity consistency.  Sincerely,  Derek McCreery | | |

## 3.4 Report 4

|  |  |
| --- | --- |
| **Christian Katsabas** <ckatsabas@gmail.com> | Thu, Mar 3, 2016 at 1:57 PM |
| To: Kristian Medri <kristian.medri@humber.ca>, Derek McCreery <derekmccreery@gmail.com> | |
| |  | | --- | | This is the Seed Germination Project's status update for the week of March 3, 2016.  Currently, the Android Application is reading dynamically from the database, and remains completed.  The web interface is functionally implemented, and is reading dynamically from the database, however it still needs to be formatted and styled to look more professional, including static text content.  Our database is fully hosted at this stage on [Hostinger.co.uk](http://hostinger.co.uk/), and should be accessible via PHP from any platform.  Currently, we are working on completing the scripts to allow our hardware to write to the database when it reads the information from our sensor suite.  Once the firmware is complete in this way, our system will be fully integrated.  For references, Derek and myself will be going through our code and hardware information, as well as our various written portions, in order to collate the entirety of what we have referenced in order to make our references section. Once complete, this section may be added to as we complete our technical report, in the event of us needing additional resources.  Sincerely,  Christian Katsabas | | |

## 3.5 Report 5

|  |  |
| --- | --- |
| **Derek McCreery** <derekmccreery@gmail.com> | Thu, Mar 17, 2016 at 2:19 PM |
| To: Kristian Medri <Kristian.Medri@humber.ca>  Cc: Christian Katsabas <ckatsabas@gmail.com> | |
| |  | | --- | | Hello Kristian this is out status update for March 17, 2016.  The website's core functionality is complete and Christian is working on getting the graphs implemented.  The Android application is complete and no modifications have been made since the last update.  The database is still hosted on [Hostinger.co.uk](http://Hostinger.co.uk) and is still functional.  Originally the plan was to read the data coming from the Arduino in Java, but I had run into a few problems with the compiler finding the correct libraries.  Instead I have decided to work around this by writing a Python script to retrieve the data, then pass it to a Java program to then send the data to the database.  Sincerely,  Derek McCreery | | |

## 3.6 Report 6

|  |  |
| --- | --- |
| **Derek McCreery** <derekmccreery@gmail.com> | Thu, Mar 31, 2016 at 2:14 PM |
| To: Kristian Medri <Kristian.Medri@humber.ca>  Cc: Christian Katsabas <ckatsabas@gmail.com> | |
| |  | | --- | | Hello Kristian,  this is our status update for the week of March 31, 2016.  The Android application had a small bug where it would not display enough data.  This issue was with the php script that the app called to return the data from the database.  The issue was fixed and the app is now displaying the correct amount of data.  The database is still hosted on [Hostinger.co.uk](http://hostinger.co.uk/) and is still functional and has received the data sent to it by the monitoring station.  The website's core functionality is complete and Christian is working on getting the graphs implemented after running into a few issues.  Sincerely,  Derek McCreery | | |

# 4. Conclusions

This system has been developed in order to meet the goal of creating a small, inexpensive system for monitoring the environment around a growing plant. It is able to measure ambient temperature, relative humidity, light intensity, and soil moisture. It then processes this data, and sends it to a database. From the database, this system is able to give this data to both a smart-phone application, and a website view, to allow for ease of access, reducing the time and manpower that would otherwise be needed to monitor these environmental variables manually. The final version of the project meets all of the requirements that were set out to be accomplished. Additional features that were introduced during the creation of the project have been considered for future revisions of the system. There are a number of additional features planned including but not limited to; supplementary lighting, water reservoir monitoring, images of current status of growth medium, and options to manually tell the system to add water to the growth medium.

# 5. Recommendations

Currently, with the hardware still in the prototype phase there are plans to reduce the cost per unit. The current cost for a single unit is quite small, only about $70 for a version without a database and a corresponding app can be made there are some ways to reduce this cost further such as buying the sensors in bulk and receiving a discount on the purchase, a decreased shipping cost associated per part and they would not be shipped individually, and the possibility of using more cost efficient materials. Code can be further refined and be made individual to a customer’s needs. A light source could be added to the growing area that would allow the plants to receive light during time where light level are not high enough, like during the night or darker days during the winter. This can be accomplished through the use of LEDs and the digital pins. Because this could give inaccurate data readings from the light sensor, it would be best to have the lights off before and during when the readings are being done. Modifications can be made to the box to allow for excess water to escape in the event that a soil moisture sensor is worn out and starts giving readings where the soil moisture is too low, despite it being at a correct amount. Another solution to this problem is the use of AC power with the sensor. Because DC power is used, slowly the gold plating from one of the prongs on the sensor will slowly lift off and travel to the other sensor. Should AC power be used, the can be avoided. The PCB files can be revised to include the connection from the 5V output to the AREF pin, so a wire isn’t needed to make the connection. Also, adjusting the orientation of the board and the connections to allow the PCD to sit on top of the Arduino and not to the side would reduce the amount of area it occupies. Some connections on the board require you to have a small wire connecting one side of the board to the other to allow the pin headers to make a proper connection. Another solution to this is by having plated-through holes.

# 6. Technical References

[1] Kone, C.T.; Hafid, A.; Boushaba, M., "Performance Management of IEEE 802.15.4 Wireless Sensor Network for Precision Agriculture," *Sensors Journal, IEEE*, vol.15, no.10, pp.5734,5747, Aug. 2015 doi: 10.1109/JSEN.2015.2442259

[2] Gutierrez Jaguey, J. ; Eng. Group, Centro de Investig. Biologicas del Noroeste, La Paz, Mexico ; Villa-Medina, J.F. ; Lopez-Guzman, A. ; Porta-Gandara, M.A., “Smartphone Irrigation Sensor,” *Sensors Journal, IEEE,* vol.15, no.9, pp.5122,5127, Sept. 2015 doi: 10.1109/JSEN.2015.2435516

[3] Gordon Henderson; Ian Jackson. (2016, February 29). WiringPi [Online]. Available: <http://www.wiringpi.com/>

[4] Jonas Gehring. (2015, June 26). Graph View - Summary & Features [Online]. Available: <http://www.android-graphview.org/>

[5] David Chester; Douglas Hunter; Silas Sewell. (2016, January 10). Rickshaw: A JavaScript toolkit for creating interactive time-series graphs [Online]. Available: <http://code.shutterstock.com/rickshaw/>

Bill Earl. (2015, May 4). Programming | Adafruit GA1A12S202 Log-scale Analog Light Sensor | Adafruit Learning System [Online]. Available: <https://learn.adafruit.com/adafruit-ga1a12s202-log-scale-analog-light-sensor/overview>

bildr. (2012, November 25). Sensing Humidity With The HIH-4030 + Arduino [Online]. Available: <http://www.bildr.org/2012/11/hih4030-arduino/>

Limor Fried. (2015, November 19). Using a Temp Sensor | TMP36 Temperature Sensor | Adafruit Learning System [Online]. Available: <https://learn.adafruit.com/tmp36-temperature-sensor/using-a-temp-sensor/>

# 7. Appendicies

## 7.1 Microcontroller Firmware

int moistPin = A0;

int tempPin = A1;

int lightPin = A2;

int humidPin = A3;

int motorPin = A5;

float moist = 0.0;

float temp = 0.0;

float light = 0.0;

float humid = 0.0;

int waterFlag=0;

float dryAmount=72.0;

float hydrateAmount=45.0;

int boardID = 1;

void setup(){

// put your setup code here, to run once:

Serial.begin(9600);

pinMode(moistPin, INPUT);//moisture

pinMode(tempPin, INPUT);//temp

pinMode(lightPin, INPUT);//light

pinMode(humidPin, INPUT);//humid

pinMode(motorPin, OUTPUT);//motor

}//end setup

void loop(){

// put your main code here, to run repeatedly:

//reset values to 0 to ensure that data being transmitted is new data (for testing purposes)

moist = 0;

temp = 0;

light = 0;

humid = 0;

waterFlag = 0;

//read data from pins and store into temp memory

moist = analogRead(moistPin);

moist = ((moist\*0.9765625)/10);//convert value ranging from 0-1023 to a number between 0-99

temp = analogRead(tempPin);

temp=((((temp\*5)/1024)-0.5)\*100);//https://learn.adafruit.com/tmp36-temperature-sensor/using-a-temp-sensor

humid = humidConv(temp);

light = lightConv();

Serial.println(moist);

delay(1000);

Serial.println(temp);

delay(1000);

Serial.println(light);

delay(1000);

Serial.println(humid);

//watering if soil moisture is lower than allowed dry amount

if(moist<=hydrateAmount)

{

digitalWrite(motorPin,HIGH);

delay(60000);

//delay(1000);

digitalWrite(motorPin,LOW);

waterFlag=1;

}

if((moist<=dryAmount)&&(waterFlag==0))

{

digitalWrite(motorPin,HIGH);

delay(20000);

//delay(500);

digitalWrite(motorPin,LOW);

delay(40000);

waterFlag=2;

}

if(waterFlag==0)

{

delay(60000);

}

//Ensures that the measuring data timing does float away over time.

/\*if(waterFlag==1){

delay(57000);}

else if(waterflag==2){

delay(15000);

}

else{

delay(57000);

}\*/

//1h between measurements

for (int x = 0; x < 59; x++)

{

for (int y = 0; y < 60; y++)

{

delay(1000);

}

}//end of 1h wait

}

//https://learn.adafruit.com/adafruit-ga1a12s202-log-scale-analog-light-sensor/use-it

float lightConv()

{

float rawRange = 1024;

float logRange = 5.0;

float data=0.0;

float value=0.0;

int raw=0;

analogReference(EXTERNAL);

for (int x=0;x<10;x++)

{

raw=analogRead(lightPin);

}

value = (raw\*logRange/rawRange);

data=pow(10,value);

analogReference(DEFAULT);

for(int x=0;x<10;x++)

{

raw=analogRead(lightPin);

}

return data;

}

float humidConv(float temp)//http://bildr.org/2012/11/hih4030-arduino/

{

int raw = analogRead(humidPin);

float supplyVolt=5.0;

float voltage = raw/1023.0\*supplyVolt;

float sensorRH = 161.0\*voltage/supplyVolt - 25.8;

float trueRH = sensorRH/(1.0546 - 0.0026\*temp);

return trueRH;

}

## 7.2 Microprocessor Communication Script

#!/usr/bin/python

import serial

import os

ser=serial.Serial('/dev/ttyACM0',9600)

while 1:

print("Soil Moisture")

moist=ser.readline()

print(moist)

print("Temperature")

temp=ser.readline()

print(temp)

print("Light")

light=ser.readline()

print(light)

print("Humidity")

humid=ser.readline()

print(humid)

#write to file here

text\_file = open("data.txt","w")

text\_file.write(moist + temp + light + humid )

text\_file.close()

os.system("java writedata");

## 7.3 Microprocessor Database Communication Program

import java.io.\*;

import java.net.\*;

public class writedata

{

public static void main(String[] args)

{

String fileName=("data.txt");

String input = null;

//https://www.caveofprogramming.com/java/java-file-reading-and-writing-files-in-java.html

try

{

FileReader fileRead=new FileReader(fileName);

BufferedReader buffRead = new BufferedReader(fileRead);

double moist=Double.parseDouble(buffRead.readLine());

System.out.println("Soil Moisture: "+moist);

double temp=Double.parseDouble(buffRead.readLine());

System.out.println("Temperature: "+temp);

double light=Double.parseDouble(buffRead.readLine());

System.out.println("Light: "+light);

double humid=Double.parseDouble(buffRead.readLine());

System.out.println("Humidity: "+humid);

String link = "http://ergoagri.esy.es/ErgoWrite.php?Temperature="+temp+"&Humidity="+humid+"&Light="+light+"&Moisture="+moist;

//test

System.out.println(link);

buffRead.close();

URL senddata = new URL(link);

URLConnection yc = senddata.openConnection();

BufferedReader in = new BufferedReader(new InputStreamReader(yc.getInputStream()));

String inputLine;

while ((inputLine = in.readLine()) != null)

System.out.println(inputLine);

in.close();

}

catch(Exception e){

System.out.println(e);

}

}

}

## 7.4 Database Input Script

<?php

$servername="mysql.hostinger.co.uk";

$username="u551669906\_admin";

$password="Kalamadea";

$dbname="u551669906\_ergo";

// Create connection

$conn = new mysqli($servername, $username, $password, $dbname);

// Check connection

if ($conn->connect\_error)

{

die("Connection failed: " . $conn->connect\_error);

}

$Temperature = $\_GET['Temperature'];

$Humidity = $\_GET['Humidity'];

$Light = $\_GET['Light'];

$Moisture = $\_GET['Moisture'];

$date= date('Y-m-d H:i:s');

$sql = "INSERT INTO DATA (Plant, Date, Temperature, Humidity, Light, Moisture)

VALUES (1,NOW(), $Temperature, $Humidity, $Light, $Moisture)";

/\*

if connection above doesn't work

$conn =mysqli\_connect($servername, $username, $password, $dbname);

if (mysqli\_query($conn,$sql)

{

echo "New record created successfully";

}

else

{

echo "Error: " . $sql . "<br>" . $conn->error;

}

\*/

if ($conn->query($sql) === TRUE)

{

echo "New record created successfully";

}

else

{

echo "Error: " . $sql . "<br>" . $conn->error;

}

$conn->close();

?>

## 7.5 Database Retrieval Script for Phone Application

<?php

/\* array for JSON response \*/

$response = array();

/\* CONNECTION SETTINGS \*/

$DB\_HOST = "mysql.hostinger.co.uk";

$DB\_UNAME = "u551669906\_admin";

$DB\_PWD = "Kalamadea";

$DB\_DATABASE = "u551669906\_ergo";

/\* Connecting to mysql database \*/

$mysqli = new mysqli($DB\_HOST, $DB\_UNAME, $DB\_PWD, $DB\_DATABASE);

if (mysqli\_connect\_errno()) {

printf("Connect failed: %s\n", mysqli\_connect\_error());

exit();

}

//$username = $\_GET['username'];

//$username = $\_POST['username'];

/\* CONSTRUCT THE QUERY change Drone to user database\*/

$query="SELECT Temperature, Humidity, Light, Moisture FROM DATA Where plant = '1' ORDER BY Date DESC LIMIT 24;";

$result = $mysqli->query($query) or die($mysqli->error.\_\_LINE\_\_);

if ($result === false) {

trigger\_error('Wrong SQL: ' . $sql . ' Error: ' . $conn->error, E\_USER\_ERROR);

} else {

$response["stuff"] = array();

while($row = $result->fetch\_assoc()) {

$stuff= array();

/\* ADD THE TABLE COLUMNS TO THE JSON OBJECT CONTENTS if it doesnt work, reverse stuff/row values\*/

$stuff["temp"] = $row['Temperature'];

$stuff["light"] = $row['Light'];

$stuff["humid"] = $row['Humidity'];

$stuff["moist"] = $row['Moisture'];

array\_push($response["stuff"], $stuff);

// $response[] = $row;

}

// success

$response["success"] = 1;

echo(json\_encode($response));

}

/\* CLOSE THE CONNECTION \*/

mysqli\_close($mysqli);

?>

## 7.6 Website Code

### 7.6.1 Main Page

<html>

<head><title>Welcome to Your ErgoAgri!</title>

</head>

<body>

<h1>Your ErgoAgri</h1>

<h2>Welcome to your ErgoAgri Web Interface, your one-stop monitoring station for all of your ErgoAgri monitoring stations!</h2>

<br/>

<br/>

<h3>Your Platforms</h3>

<p><a href="./Plant\_One.php">Platform 1</a></p>

<h3>Notifications</h3>

<p><b>Platform 1:</b>

<?php

$servername="mysql.hostinger.co.uk";

$username="u551669906\_admin";

$password="Kalamadea";

$dbname="u551669906\_ergo";

$query="SELECT Date,Moisture FROM `DATA` WHERE Plant=1 ORDER by Date";

$indicies="SELECT COUNT(\*) FROM `DATA` WHERE Plant=1 ";

$dates=array();

$moists=array();

$counter = 0;

$conn=new mysqli($servername,$username,$password,$dbname);

if(mysqli\_connect\_errno($con)){

die("Connection Failed: " . mysqli\_connect\_error);

}

$counts=mysqli\_query($conn,$indicies) or die(mysql\_error());

$entries=mysqli\_fetch\_row($counts);

if($result=mysqli\_query($conn,$query))

{

while($row=mysqli\_fetch\_array($result))

{

$dates[]=$row['Date'];

$moists[]=$row['Moisture'];

}

mysqli\_free\_result($result);

}

mysqli\_close($conn);

if(end($moists)<=70)

{

echo "<font color='red'>Recently needed watering (below 70% moisture)!</font></p>";

}

else

{

echo "<font color='green'>Moisture levels OK!</font></p>";

}

?>

</body>

</html>

### 7.6.2 Station Page

<?php

echo "i got here<br>";

$servername="mysql.hostinger.co.uk";

$username="u551669906\_admin";

$password="Kalamadea";

$dbname="u551669906\_ergo";

$date\_get="SELECT TO\_CHAR(Date) FROM `DATA` WHERE Plant=1 ORDER by Date";

$temp\_get="SELECT Temperature FROM `DATA` WHERE Plant=1 ORDER by Date";

$hum\_get="SELECT Humidity FROM `DATA` WHERE Plant=1 ORDER by Date";

$light\_get="SELECT Light FROM `DATA` WHERE Plant=1 ORDER by Date";

$moist\_get="SELECT Moisture FROM `DATA` WHERE Plant=1 ORDER by Date";

$query="SELECT Date,Temperature,Humidity,Light,Moisture FROM `DATA` WHERE Plant=1 ORDER by Date";

$indicies="SELECT COUNT(\*) FROM `DATA` WHERE Plant=1 ";

$dates=array();

$temps=array();

$hums=array();

$lights=array();

$moists=array();

// $queries = array($date\_get,$temp\_get,$hum\_get,$light\_get,$moist\_get);

// $arrays= array($dates,$temps,$hums,$lights,$moists);

$counter = 0;

$conn=new mysqli($servername,$username,$password,$dbname);

if(mysqli\_connect\_errno($con)){

die("Connection Failed: " . mysqli\_connect\_error);

}

echo "I connected<br>";

$counts=mysqli\_query($conn,$indicies) or die(mysql\_error());

$thing=$counts->fetch\_row();

if($result=mysqli\_query($conn,$query))

{

while($row=mysqli\_fetch\_array($result))

{

$dates[]=$row['Date'];

$temps[]=$row['Temperature'];

$hums[]=$row['Humidity'];

$lights[]=$row['Light'];

$moists[]=$row['Moisture'];

}

mysqli\_free\_result($result);

}

mysqli\_close($conn);

echo "Connection closed<br>";

echo $thing[0];

echo "<br>";

echo $temps[0];

echo "<br>";

$temp\_no=count($temps);

$date\_no=count($dates);

for($counter=0;$counter<$date\_no;$counter++)

{

echo $dates[$counter];

echo "<br>";

}

echo "<br>";

for($counter=0;$counter<$entries[0];$counter++)

{

$temp\_list = $dates[$counter] + ' - ' + $temps[$counter] + '\n';

}

echo "$temp\_list";

?>

### 7.6.3 Humidity Page

<html>

<head><title>ErgoAgri - Platform 1 - Humidity</title>

</head>

<body>

<?php

$servername="mysql.hostinger.co.uk";

$username="u551669906\_admin";

$password="Kalamadea";

$dbname="u551669906\_ergo";

$query="SELECT Date,Humidity FROM `DATA` WHERE Plant=1 ORDER by Date";

$indicies="SELECT COUNT(\*) FROM `DATA` WHERE Plant=1 ";

$dates=array();

$hums=array();

$counter = 0;

$conn=new mysqli($servername,$username,$password,$dbname);

if(mysqli\_connect\_errno($con)){

die("Connection Failed: " . mysqli\_connect\_error);

}

$counts=mysqli\_query($conn,$indicies) or die(mysql\_error());

$entries=mysqli\_fetch\_row($counts);

if($result=mysqli\_query($conn,$query))

{

while($row=mysqli\_fetch\_array($result))

{

$dates[]=$row['Date'];

$hums[]=$row['Humidity'];

}

$dateList = $dates;

mysqli\_free\_result($result);

}

mysqli\_close($conn);

for($counter=$entries[0];$counter>0;$counter--)

{

if($dates[$counter])

{

if($counter!=0){

$hum\_list .= $dates[$counter];}

else{

$hum\_list = $dates[$counter];}

$hum\_list .= " - ";

$hum\_list .= $hums[$counter];

$hum\_list .= "%<br>";

}

}

?>

<h1>Humidity for Plant 1</h1>

<a href="../Plant\_One.php">Back to Overview</a>

<p>

<script type="text/javascript">

var datesList = <?php echo json\_encode($dateList);?>;

var humsList = <?php echo json\_encode($hums);?>;

</script>

<link type="text/css" rel="stylesheet" href="./Rickshaw/rickshaw.min.css">

<script src="./Rickshaw/vendor/d3.min.js"></script>

<script src="./Rickshaw/vendor/d3.layout.min.js"></script>

<script src="./Rickshaw/rickshaw.min.js"></script>

<style>

#chart\_container {

position: relative;

font-family: Arial, Helvetica, sans-serif;

}

#chart {

position: relative;

left: 40px;

}

#y\_axis {

position: absolute;

top: 0;

bottom: 0;

width: 40px;

}

</style>

<div id="chart\_container">

<div id="y\_axis"></div>

<div id="chart"></div>

</div>

<script type="text/javascript">

var graph = new Rickshaw.Graph( {

element: document.querySelector("#chart"),

width: 800,

height: 400,

renderer: 'line',

interpolation: 'linear',

series: [ {

data: [{x: getDatePoint(0), y: getHum(0)},{x: getDatePoint(1), y: getHum(1)},{x: getDatePoint(2), y: getHum(2)},{x: getDatePoint(3), y: getHum(3)},{x: getDatePoint(4), y: getHum(4)},{x: getDatePoint(5), y: getHum(5)},{x: getDatePoint(6), y: getHum(6)},{x: getDatePoint(7), y: getHum(7)},{x: getDatePoint(8), y: getHum(8)},{x: getDatePoint(9), y: getHum(9)},{x: getDatePoint(10), y: getHum(10)},{x: getDatePoint(11), y: getHum(11)},{x: getDatePoint(12), y: getHum(12)},{x: getDatePoint(13), y: getHum(13)},{x: getDatePoint(14), y: getHum(14)},{x: getDatePoint(15), y: getHum(15)},{x: getDatePoint(16), y: getHum(16)},{x: getDatePoint(17), y: getHum(17)},{x: getDatePoint(18), y: getHum(18)},{x: getDatePoint(19), y: getHum(19)},{x: getDatePoint(20), y: getHum(20)},{x: getDatePoint(21), y: getHum(21)},{x: getDatePoint(22), y: getHum(22)},{x: getDatePoint(23), y: getHum(23)}],

color: "steelblue",

name: "Humidity"

} ]

} );

var y\_axis = new Rickshaw.Graph.Axis.Y( {

graph: graph,

orientation: "left",

tickFormat: Rickshaw.Fixtures.Number.formatKMBT,

element: document.getElementById("y\_axis")

} );

var hoverDetail = new Rickshaw.Graph.HoverDetail( {

graph: graph,

xFormatter: function(x) {

return new Date(x \* 1000).toString();

}

} );

var x\_axis = new Rickshaw.Graph.Axis.X({

graph: graph,

pixelsPerTick: 175,

tickFormat: function(x)

{

return stringer = new Date(x\*1000).toLocaleString()

}

})

x\_axis.render();

graph.render();

function getDatePoint(count)

{

return parseInt(getDate(datesList[(datesList.length)-24+count]));

}

function getHum(count)

{

return parseFloat(humsList[humsList.length-24+count]);

}

function getDate(datestring)

{

var parts = datestring.match(/(\d{4})-(\d{2})-(\d{2}) (\d{2}):(\d{2}):(\d{2})/);

return parseInt((Date.UTC(+parts[1], +parts[2]-1, +parts[3], +parts[4], +parts[5], +parts[6]))/1000);

}

</script>

</p>

<p><?php echo $hum\_list ?></p>

</body>

</html>

### 7.6.4 Light Page

<html>

<head><title>ErgoAgri - Platform 1 - Light Levels</title>

</head>

<body>

<?php

$servername="mysql.hostinger.co.uk";

$username="u551669906\_admin";

$password="Kalamadea";

$dbname="u551669906\_ergo";

$query="SELECT Date,Light FROM `DATA` WHERE Plant=1 ORDER by Date";

$indicies="SELECT COUNT(\*) FROM `DATA` WHERE Plant=1 ";

$dates=array();

$lights=array();

$counter = 0;

$conn=new mysqli($servername,$username,$password,$dbname);

if(mysqli\_connect\_errno($con)){

die("Connection Failed: " . mysqli\_connect\_error);

}

$counts=mysqli\_query($conn,$indicies) or die(mysql\_error());

$entries=mysqli\_fetch\_row($counts);

if($result=mysqli\_query($conn,$query))

{

while($row=mysqli\_fetch\_array($result))

{

$dates[]=$row['Date'];

$lights[]=$row['Light'];

}

$dateList = $dates;

mysqli\_free\_result($result);

}

mysqli\_close($conn);

for($counter=$entries[0];$counter>0;$counter--)

{

if($dates[$counter])

{

if($counter!=0){

$light\_list .= $dates[$counter];}

else{

$light\_list = $dates[$counter];}

$light\_list .= " - ";

$light\_list .= $lights[$counter];

$light\_list .= " lux<br>";

}

}

?>

<h1>Light Levels for Plant 1</h1>

<a href="../Plant\_One.php">Back to Overview</a>

<p>

<script type="text/javascript">

var datesList = <?php echo json\_encode($dateList);?>;

var lightsList = <?php echo json\_encode($lights);?>;

</script>

<link type="text/css" rel="stylesheet" href="./Rickshaw/rickshaw.min.css">

<script src="./Rickshaw/vendor/d3.min.js"></script>

<script src="./Rickshaw/vendor/d3.layout.min.js"></script>

<script src="./Rickshaw/rickshaw.min.js"></script>

<style>

#chart\_container {

position: relative;

font-family: Arial, Helvetica, sans-serif;

}

#chart {

position: relative;

left: 40px;

}

#y\_axis {

position: absolute;

top: 0;

bottom: 0;

width: 40px;

}

</style>

<div id="chart\_container">

<div id="y\_axis"></div>

<div id="chart"></div>

</div>

<script type="text/javascript">

var graph = new Rickshaw.Graph( {

element: document.querySelector("#chart"),

width: 800,

height: 400,

renderer: 'line',

interpolation: 'linear',

series: [ {

data: [{x: getDatePoint(0), y: getLight(0)},{x: getDatePoint(1), y: getLight(1)},{x: getDatePoint(2), y: getLight(2)},{x: getDatePoint(3), y: getLight(3)},{x: getDatePoint(4), y: getLight(4)},{x: getDatePoint(5), y: getLight(5)},{x: getDatePoint(6), y: getLight(6)},{x: getDatePoint(7), y: getLight(7)},{x: getDatePoint(8), y: getLight(8)},{x: getDatePoint(9), y: getLight(9)},{x: getDatePoint(10), y: getLight(10)},{x: getDatePoint(11), y: getLight(11)},{x: getDatePoint(12), y: getLight(12)},{x: getDatePoint(13), y: getLight(13)},{x: getDatePoint(14), y: getLight(14)},{x: getDatePoint(15), y: getLight(15)},{x: getDatePoint(16), y: getLight(16)},{x: getDatePoint(17), y: getLight(17)},{x: getDatePoint(18), y: getLight(18)},{x: getDatePoint(19), y: getLight(19)},{x: getDatePoint(20), y: getLight(20)},{x: getDatePoint(21), y: getLight(21)},{x: getDatePoint(22), y: getLight(22)},{x: getDatePoint(23), y: getLight(23)}],

color: "steelblue",

name: "Light Level"

} ]

} );

var y\_axis = new Rickshaw.Graph.Axis.Y( {

graph: graph,

orientation: "left",

tickFormat: Rickshaw.Fixtures.Number.formatKMBT,

element: document.getElementById("y\_axis")

} );

var hoverDetail = new Rickshaw.Graph.HoverDetail( {

graph: graph,

xFormatter: function(x) {

return new Date(x \* 1000).toString();

}

} );

var x\_axis = new Rickshaw.Graph.Axis.X({

graph: graph,

pixelsPerTick: 175,

tickFormat: function(x)

{

return stringer = new Date(x\*1000).toLocaleString()

}

})

x\_axis.render();

graph.render();

function getDatePoint(count)

{

return parseInt(getDate(datesList[(datesList.length)-24+count]));

}

function getLight(count)

{

return parseFloat(lightsList[lightsList.length-24+count]);

}

function getDate(datestring)

{

var parts = datestring.match(/(\d{4})-(\d{2})-(\d{2}) (\d{2}):(\d{2}):(\d{2})/);

return parseInt((Date.UTC(+parts[1], +parts[2]-1, +parts[3], +parts[4], +parts[5], +parts[6]))/1000);

}

</script>

</p>

<p><?php echo $light\_list ?></p>

</body>

</html>

### 7.6.5 Temperature Page

<html>

<head><title>ErgoAgri - Platform 1 - Temperature</title>

</head>

<body>

<?php

$servername="mysql.hostinger.co.uk";

$username="u551669906\_admin";

$password="Kalamadea";

$dbname="u551669906\_ergo";

$query="SELECT Date,Temperature FROM `DATA` WHERE Plant=1 ORDER by Date";

$indicies="SELECT COUNT(\*) FROM `DATA` WHERE Plant=1 ";

$dates=array();

$temps=array();

$counter = 0;

$conn=new mysqli($servername,$username,$password,$dbname);

if(mysqli\_connect\_errno($con)){

die("Connection Failed: " . mysqli\_connect\_error);

}

$counts=mysqli\_query($conn,$indicies) or die(mysql\_error());

$entries=mysqli\_fetch\_row($counts);

if($result=mysqli\_query($conn,$query))

{

while($row=mysqli\_fetch\_array($result))

{

$dates[]=$row['Date'];

$temps[]=$row['Temperature'];

}

$dateList = $dates;

mysqli\_free\_result($result);

}

mysqli\_close($conn);

for($counter=$entries[0];$counter>0;$counter--)

{

if($dates[$counter])

{

if($counter!=$entries[0]){

$temp\_list .= $dates[$counter];}

else{

$temp\_list = $dates[$counter];}

$temp\_list .= " - ";

$temp\_list .= $temps[$counter];

$temp\_list .= "&deg";

$temp\_list .="C";

$temp\_list .= "<br>";

}

}

?>

<h1>Temperature for Plant 1</h1>

<a href="../Plant\_One.php">Back to Overview</a>

<p>

<script type="text/javascript">

var datesList = <?php echo json\_encode($dateList);?>;

var tempsList = <?php echo json\_encode($temps);?>;

</script>

<link type="text/css" rel="stylesheet" href="./Rickshaw/rickshaw.min.css">

<script src="./Rickshaw/vendor/d3.min.js"></script>

<script src="./Rickshaw/vendor/d3.layout.min.js"></script>

<script src="./Rickshaw/rickshaw.min.js"></script>

<style>

#chart\_container {

position: relative;

font-family: Arial, Helvetica, sans-serif;

}

#chart {

position: relative;

left: 40px;

}

#y\_axis {

position: absolute;

top: 0;

bottom: 0;

width: 40px;

}

</style>

<div id="chart\_container">

<div id="y\_axis"></div>

<div id="chart"></div>

</div>

<script type="text/javascript">

var graph = new Rickshaw.Graph( {

element: document.querySelector("#chart"),

width: 900,

height: 400,

renderer: 'line',

interpolation: 'linear',

series: [ {

data: [{x: getDatePoint(0), y: getTemp(0)},{x: getDatePoint(1), y: getTemp(1)},{x: getDatePoint(2), y: getTemp(2)},{x: getDatePoint(3), y: getTemp(3)},{x: getDatePoint(4), y: getTemp(4)},{x: getDatePoint(5), y: getTemp(5)},{x: getDatePoint(6), y: getTemp(6)},{x: getDatePoint(7), y: getTemp(7)},{x: getDatePoint(8), y: getTemp(8)},{x: getDatePoint(9), y: getTemp(9)},{x: getDatePoint(10), y: getTemp(10)},{x: getDatePoint(11), y: getTemp(11)},{x: getDatePoint(12), y: getTemp(12)},{x: getDatePoint(13), y: getTemp(13)},{x: getDatePoint(14), y: getTemp(14)},{x: getDatePoint(15), y: getTemp(15)},{x: getDatePoint(16), y: getTemp(16)},{x: getDatePoint(17), y: getTemp(17)},{x: getDatePoint(18), y: getTemp(18)},{x: getDatePoint(19), y: getTemp(19)},{x: getDatePoint(20), y: getTemp(20)},{x: getDatePoint(21), y: getTemp(21)},{x: getDatePoint(22), y: getTemp(22)},{x: getDatePoint(23), y: getTemp(23)}],

color: "steelblue",

name: "Temperature"

} ]

} );

var y\_axis = new Rickshaw.Graph.Axis.Y( {

graph: graph,

orientation: "left",

tickFormat: Rickshaw.Fixtures.Number.formatKMBT,

element: document.getElementById("y\_axis")

} );

var hoverDetail = new Rickshaw.Graph.HoverDetail( {

graph: graph,

xFormatter: function(x) {

return new Date(x \* 1000).toString();

}

} );

var x\_axis = new Rickshaw.Graph.Axis.X({

graph: graph,

pixelsPerTick: 175,

tickFormat: function(x)

{

return stringer = new Date(x\*1000).toLocaleString()

}

})

x\_axis.render();

graph.render();

function getDatePoint(count)

{

return parseInt(getDate(datesList[(datesList.length)-24+count]));

}

function getTemp(count)

{

return parseFloat(tempsList[tempsList.length-24+count]);

}

function getDate(datestring)

{

var parts = datestring.match(/(\d{4})-(\d{2})-(\d{2}) (\d{2}):(\d{2}):(\d{2})/);

return parseInt((Date.UTC(+parts[1], +parts[2]-1, +parts[3], +parts[4], +parts[5], +parts[6]))/1000);

}

</script>

</p>

<p><?php echo $temp\_list ?></p>

</body>

</html>

### 7.6.6 Moisture Page

<html>

<head><title>ErgoAgri - Platform 1 - Moisture</title>

</head>

<body>

<?php

$servername="mysql.hostinger.co.uk";

$username="u551669906\_admin";

$password="Kalamadea";

$dbname="u551669906\_ergo";

$query="SELECT Date,Moisture FROM `DATA` WHERE Plant=1 ORDER by Date";

$indicies="SELECT COUNT(\*) FROM `DATA` WHERE Plant=1 ";

$dates=array();

$moists=array();

$counter = 0;

$conn=new mysqli($servername,$username,$password,$dbname);

if(mysqli\_connect\_errno($con)){

die("Connection Failed: " . mysqli\_connect\_error);

}

$counts=mysqli\_query($conn,$indicies) or die(mysql\_error());

$entries=mysqli\_fetch\_row($counts);

if($result=mysqli\_query($conn,$query))

{

while($row=mysqli\_fetch\_array($result))

{

$dates[]=$row['Date'];

$moists[]=$row['Moisture'];

}

$dateList = $dates;

mysqli\_free\_result($result);

}

mysqli\_close($conn);

for($counter=$entries[0];$counter>0;$counter--)

{

if($dates[$counter])

{

if($counter!=0){

$moist\_list .= $dates[$counter];}

else{

$moist\_list = $dates[$counter];}

$moist\_list .= " - ";

$moist\_list .= $moists[$counter];

$moist\_list .= "%<br>";

}

}

?>

<h1>Moisture Levels for Plant 1</h1>

<a href="../Plant\_One.php">Back to Overview</a>

<p>

<script type="text/javascript">

var datesList = <?php echo json\_encode($dateList);?>;

var moistsList = <?php echo json\_encode($moists);?>;

</script>

<link type="text/css" rel="stylesheet" href="./Rickshaw/rickshaw.min.css">

<script src="./Rickshaw/vendor/d3.min.js"></script>

<script src="./Rickshaw/vendor/d3.layout.min.js"></script>

<script src="./Rickshaw/rickshaw.min.js"></script>

<style>

#chart\_container {

position: relative;

font-family: Arial, Helvetica, sans-serif;

}

#chart {

position: relative;

left: 40px;

}

#y\_axis {

position: absolute;

top: 0;

bottom: 0;

width: 40px;

}

</style>

<div id="chart\_container">

<div id="y\_axis"></div>

<div id="chart"></div>

</div>

<script type="text/javascript">

var graph = new Rickshaw.Graph( {

element: document.querySelector("#chart"),

width: 800,

height: 400,

renderer: 'line',

interpolation: 'linear',

series: [ {

data: [{x: getDatePoint(0), y: getMoist(0)},{x: getDatePoint(1), y: getMoist(1)},{x: getDatePoint(2), y: getMoist(2)},{x: getDatePoint(3), y: getMoist(3)},{x: getDatePoint(4), y: getMoist(4)},{x: getDatePoint(5), y: getMoist(5)},{x: getDatePoint(6), y: getMoist(6)},{x: getDatePoint(7), y: getMoist(7)},{x: getDatePoint(8), y: getMoist(8)},{x: getDatePoint(9), y: getMoist(9)},{x: getDatePoint(10), y: getMoist(10)},{x: getDatePoint(11), y: getMoist(11)},{x: getDatePoint(12), y: getMoist(12)},{x: getDatePoint(13), y: getMoist(13)},{x: getDatePoint(14), y: getMoist(14)},{x: getDatePoint(15), y: getMoist(15)},{x: getDatePoint(16), y: getMoist(16)},{x: getDatePoint(17), y: getMoist(17)},{x: getDatePoint(18), y: getMoist(18)},{x: getDatePoint(19), y: getMoist(19)},{x: getDatePoint(20), y: getMoist(20)},{x: getDatePoint(21), y: getMoist(21)},{x: getDatePoint(22), y: getMoist(22)},{x: getDatePoint(23), y: getMoist(23)}],

color: "steelblue",

name: "Moisture"

} ]

} );

var y\_axis = new Rickshaw.Graph.Axis.Y( {

graph: graph,

orientation: "left",

tickFormat: Rickshaw.Fixtures.Number.formatKMBT,

element: document.getElementById("y\_axis")

} );

var hoverDetail = new Rickshaw.Graph.HoverDetail( {

graph: graph,

xFormatter: function(x) {

return new Date(x \* 1000).toString();

}

} );

graph.render();

function getDatePoint(count)

{

return parseInt(getDate(datesList[(datesList.length)-24+count]));

}

function getMoist(count)

{

return parseFloat(moistsList[moistsList.length-24+count]);

}

function getDate(datestring)

{

var parts = datestring.match(/(\d{4})-(\d{2})-(\d{2}) (\d{2}):(\d{2}):(\d{2})/);

return parseInt((Date.UTC(+parts[1], +parts[2]-1, +parts[3], +parts[4], +parts[5], +parts[6]))/1000);

}

</script>

</p>

<p><?php echo $moist\_list ?></p>

</body>

</html>

## 7.7 Android Phone Application

### 7.7.1 Main Menu

package com.example.chris.ergoagri;

import android.app.Activity;

import android.app.ProgressDialog;

import android.content.Context;

import android.content.Intent;

import android.content.pm.ActivityInfo;

import android.content.res.Configuration;

import android.os.AsyncTask;

import android.os.Bundle;

import android.os.Environment;

import android.util.Log;

import android.view.Menu;

import android.view.MenuItem;

import android.view.View;

import android.widget.Button;

import android.widget.Toast;

import org.apache.http.HttpEntity;

import org.apache.http.HttpResponse;

import org.apache.http.client.HttpClient;

import org.apache.http.client.methods.HttpGet;

import org.apache.http.impl.client.DefaultHttpClient;

import org.apache.http.util.EntityUtils;

import org.json.JSONArray;

import org.json.JSONException;

import org.json.JSONObject;

import java.io.BufferedReader;

import java.io.File;

import java.io.FileOutputStream;

import java.io.FileReader;

import java.io.InputStreamReader;

import java.io.OutputStreamWriter;

import java.net.HttpURLConnection;

import java.net.URI;

import java.net.URL;

import java.util.Scanner;

public class MainMenu extends Activity {

int[] date;

double[] temp;

double[] light;

double[] humid;

double[] moist;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main\_menu);

if (tablet()) {

setRequestedOrientation(ActivityInfo.SCREEN\_ORIENTATION\_LANDSCAPE);

}

//Giving a default array of values, mainly used for testing to ensure that data was updated

date=new int[]{ 1, 2, 3, 4, 5, 6, 7, 8, 9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24};

temp=new double[]{ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0};

light=new double[]{ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0};

humid=new double[]{ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0};

moist=new double[]{ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0};

newUpdate();

}

//go to Current Data screen

public void gotoCurrent(View view)

{

Intent intent = new Intent(MainMenu.this, CurrentData.class);

intent.putExtra("date",date);

intent.putExtra("temp",temp);

intent.putExtra("humid",humid);

intent.putExtra("light",light);

intent.putExtra("moist", moist);

MainMenu.this.startActivity(intent);

}

//go to Temperature History screen, and from there are able to continue to the graph.

public void gotoTemp(View view)

{

Intent intent = new Intent(MainMenu.this, TempHistory.class);

intent.putExtra("date",date);

intent.putExtra("temp",temp);

MainMenu.this.startActivity(intent);

}

//go to Light History screen, and from there are able to continue to the graph.

public void gotoLight(View view)

{

Intent intent = new Intent(MainMenu.this,LightHistory.class);

intent.putExtra("date",date);

intent.putExtra("light",light);

MainMenu.this.startActivity(intent);

}

//go to Moisture History screen, and from there are able to continue to the graph.

public void gotoMoisture(View view)

{

Intent intent = new Intent(MainMenu.this,MoistureHistory.class);

intent.putExtra("date",date);

intent.putExtra("moist",moist);

MainMenu.this.startActivity(intent);

}

//go to Humidity History screen, and from there are able to continue to the graph.

public void gotoHumid(View view)

{

Intent intent = new Intent(MainMenu.this,HumidHistory.class);

intent.putExtra("date",date);

intent.putExtra("humid",humid);

MainMenu.this.startActivity(intent);

}

//This is the function that runs in the oncreate to sync the local database with the one on the server.

public void newUpdate()

{

new LoadDataActivity(getBaseContext(), 0).execute();

}

private class LoadDataActivity extends AsyncTask<String, Void, String> {

private Context context;

private int byGetOrPost = 0;

JSONArray stuff = null;

private static final String TAG\_JSONNAME = "stuff";

private static final String TAG\_TEMP = "temp";

private static final String TAG\_LIGHT = "light";

private static final String TAG\_HUMID = "humid";

private static final String TAG\_MOIST = "moist";

//flag 0 means get and 1 means post.(By default it is get.)

public LoadDataActivity(Context context, int flag) {

this.context = context;

byGetOrPost = flag;

}

//Creates a progressdialog so the user knows the app is performing actions.

protected void onPreExecute() {}

//Connects to database and recieves a JSONArray

@Override

protected String doInBackground(String... arg0) {

if (byGetOrPost == 0) { //means by Get Method

try {

String link = "http://ergoagri.esy.es/ErgoData.php"; //makes this your url to the php script

link = link.replaceAll(" ", "%20");

URL url = new URL(link);

HttpClient client = new DefaultHttpClient();

HttpGet request = new HttpGet();

request.setURI(new URI(link));

HttpResponse response = client.execute(request);

HttpEntity httpEntity = response.getEntity();

String myResponse = EntityUtils.toString(httpEntity);

// Making a request to url and getting response

Log.d("Response: ", "> " + myResponse);

if (myResponse != null) {

try {

JSONObject jsonObj = new JSONObject(myResponse);

// Getting JSON Array node

stuff = jsonObj.getJSONArray(TAG\_JSONNAME);

} catch (JSONException e) {

e.printStackTrace();

}

} else {

Log.e("ServiceHandler", "Couldn't get any data from the url");

}

return null;

} catch (Exception e) {

return "Exception: " + e.getMessage();

}

} else {

return "False";

}

}

//Iterates through the data gathered from the return string and displays it.

@Override

protected void onPostExecute(String result) {

try {

// looping through All Contacts

for (int i = 0; i < stuff.length(); i++) {

JSONObject c = stuff.getJSONObject(i);

temp[i] = c.getDouble(TAG\_TEMP);

light[i] = c.getDouble(TAG\_LIGHT);

humid[i] = c.getDouble(TAG\_HUMID);

moist[i] = c.getDouble(TAG\_MOIST);

}

} catch (JSONException e) {

e.printStackTrace();

}

}

}

public void autoupdate()

{

try {

Toast.makeText(MainMenu.this, "Loading Database", Toast.LENGTH\_SHORT).show();

new Thread() {

@Override

public void run() {

try {

URL url = new URL("http://phdeats.esy.es/ergodata.txt");

HttpURLConnection urlConnection;

File sdcard = Environment.getExternalStorageDirectory();

File dir = new File(sdcard.getAbsolutePath() + "/tmp/");

dir.mkdir();

File remoteFile = new File(dir, "data.txt");

String StringBuffer2=new String();

//File remoteFile = new File(path + saveTo);

if (remoteFile.exists()) {

try {

//url = new URL(toDownload);

urlConnection = (HttpURLConnection) url.openConnection();

int responceCode = urlConnection.getResponseCode();

if (responceCode == 200) {

BufferedReader bufferedReader = new BufferedReader(new InputStreamReader(url.openStream()));

String StringBuffer;

FileOutputStream outStream = new FileOutputStream(remoteFile);

OutputStreamWriter osw = new OutputStreamWriter(outStream);

while ((StringBuffer = bufferedReader.readLine()) != null) {

//osw.append(StringBuffer + "\n");

StringBuffer2 += StringBuffer + "\n";

}

osw.write(StringBuffer2);

osw.close();

outStream.close();

bufferedReader.close();

//outStream.close();

//osw.close();

}

} catch (Exception e) {

e.printStackTrace();

}

}

try {

//File sdcard = Environment.getExternalStorageDirectory();

//File dir = new File(sdcard.getAbsolutePath() + "/tmp/");

File fileread = new File(dir, "data.txt");

StringBuilder text = new StringBuilder();

BufferedReader br = new BufferedReader(new FileReader(fileread));

int x=0;

int arrayindex=0;

String line;

while ((line = br.readLine()) != null) {

text.append(line);

text.append('\n');

}

br.close();

Scanner scan = new Scanner(text.toString());

scan.useDelimiter("\n");

while (arrayindex<20) {

if(x==0)

{

date[arrayindex]=Integer.parseInt(scan.next());

x=1;

}

if (x==1)

{

temp[arrayindex]=Integer.parseInt(scan.next());

x=2;

}

if (x==2)

{

light[arrayindex]=Integer.parseInt(scan.next());

x=3;

}

if (x==3)

{

humid[arrayindex]=Integer.parseInt(scan.next());

x=4;

}

if (x==4)

{

moist[arrayindex]=Integer.parseInt(scan.next());

arrayindex++;

x=0;

}

}

br.close();

}

catch (Exception e) {

e.printStackTrace();

}

}

catch (Exception e)

{

e.printStackTrace();

}

}

}.start();

}catch (Exception e) {

e.printStackTrace();

}

}

//This function is called when the user presses the update button and runs the update function

public void update(View view) {

newUpdate();

}

//Checks to see if the device is a tablet and returns true if it is

private boolean tablet() {

return (this.getResources().getConfiguration().screenLayout

& Configuration.SCREENLAYOUT\_SIZE\_MASK)

>= Configuration.SCREENLAYOUT\_SIZE\_LARGE;

}

@Override

public boolean onCreateOptionsMenu(Menu menu) {

// Inflate the menu; this adds items to the action bar if it is present.

getMenuInflater().inflate(R.menu.menu\_main\_menu, menu);

return true;

}

@Override

public boolean onOptionsItemSelected(MenuItem item) {

// Handle action bar item clicks here. The action bar will

// automatically handle clicks on the Home/Up button, so long

// as you specify a parent activity in AndroidManifest.xml.

int id = item.getItemId();

//noinspection SimplifiableIfStatement

if (id == R.id.action\_settings) {

return true;

}

return super.onOptionsItemSelected(item);

}

}

### 7.7.2 Current Data

package com.example.chris.ergoagri;

import android.app.Activity;

import android.content.Intent;

import android.content.pm.ActivityInfo;

import android.content.res.Configuration;

import android.os.Bundle;

import android.view.Menu;

import android.view.MenuItem;

import android.widget.TextView;

public class CurrentData extends Activity {

double[] temp;

double[] humid;

double[] light;

double[] moist;

String tempdata;

String humiddata;

String lightdata;

String moistdata;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_current\_data);

if (tablet()) {

setRequestedOrientation(ActivityInfo.SCREEN\_ORIENTATION\_LANDSCAPE);

}

TextView temptext = (TextView) findViewById(R.id.curTemp);

TextView humidtext = (TextView) findViewById(R.id.curHumid);

TextView lighttext = (TextView) findViewById(R.id.curLight);

TextView moisttext = (TextView) findViewById(R.id.curMoist);

Intent i = getIntent();

temp=i.getDoubleArrayExtra("temp");

humid=i.getDoubleArrayExtra("humid");

light=i.getDoubleArrayExtra("light");

moist=i.getDoubleArrayExtra("moist");

//the data at index 0 is the most recent entry in the database, so I pull these records out and display them to the user.

tempdata=temp[0]+ "°C";

temptext.setText(tempdata);

humiddata=humid[0]+ "%";

humidtext.setText(humiddata);

lightdata=light[0]+" ";

lighttext.setText(lightdata);

moistdata=moist[0]+ "%";

moisttext.setText(moistdata);

}

//Checks to see if the device is a tablet and returns true if it is

private boolean tablet() {

return (this.getResources().getConfiguration().screenLayout

& Configuration.SCREENLAYOUT\_SIZE\_MASK)

>= Configuration.SCREENLAYOUT\_SIZE\_LARGE;

}

@Override

public boolean onCreateOptionsMenu(Menu menu) {

// Inflate the menu; this adds items to the action bar if it is present.

getMenuInflater().inflate(R.menu.menu\_current\_data, menu);

return true;

}

@Override

public boolean onOptionsItemSelected(MenuItem item) {

// Handle action bar item clicks here. The action bar will

// automatically handle clicks on the Home/Up button, so long

// as you specify a parent activity in AndroidManifest.xml.

int id = item.getItemId();

//noinspection SimplifiableIfStatement

if (id == R.id.action\_settings) {

return true;

}

return super.onOptionsItemSelected(item);

}

}

### 7.7.3 Humidity History

package com.example.chris.ergoagri;

import android.app.Activity;

import android.content.Intent;

import android.content.IntentFilter;

import android.content.pm.ActivityInfo;

import android.content.res.Configuration;

import android.support.v7.app.AppCompatActivity;

import android.os.Bundle;

import android.text.method.ScrollingMovementMethod;

import android.view.Gravity;

import android.view.Menu;

import android.view.MenuItem;

import android.view.View;

import android.widget.TextView;

public class HumidHistory extends Activity {

int[] date;

double[] humid;

String display;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_humid\_history);

if (tablet()) {

setRequestedOrientation(ActivityInfo.SCREEN\_ORIENTATION\_LANDSCAPE);

}

Intent i = getIntent();

date=i.getIntArrayExtra("date");

humid=i.getDoubleArrayExtra("humid");

TextView text = (TextView) findViewById(R.id.textViewHumid2);

text.setMovementMethod(new ScrollingMovementMethod());

display = (date[0]) + "\t\t\t\t" + humid[0] + "%" + "\n";

for(int x=1;x<date.length;x++)

{

display += (date[x])+ "\t\t\t\t" + humid[x] + "%" + "\n";

}

text.setText(display);

}

//Checks to see if the device is a tablet and returns true if it is

private boolean tablet() {

return (this.getResources().getConfiguration().screenLayout

& Configuration.SCREENLAYOUT\_SIZE\_MASK)

>= Configuration.SCREENLAYOUT\_SIZE\_LARGE;

}

public void gotoHumidGraph(View view)

{

Intent intent = new Intent(HumidHistory.this,HumidGraph.class);

intent.putExtra("date",date);

intent.putExtra("humid",humid);

HumidHistory.this.startActivity(intent);

}

@Override

public boolean onCreateOptionsMenu(Menu menu) {

// Inflate the menu; this adds items to the action bar if it is present.

getMenuInflater().inflate(R.menu.menu\_humid\_history, menu);

return true;

}

@Override

public boolean onOptionsItemSelected(MenuItem item) {

// Handle action bar item clicks here. The action bar will

// automatically handle clicks on the Home/Up button, so long

// as you specify a parent activity in AndroidManifest.xml.

int id = item.getItemId();

//noinspection SimplifiableIfStatement

if (id == R.id.action\_settings) {

return true;

}

return super.onOptionsItemSelected(item);

}

}

### 7.7.4 Humidity Graph

package com.example.chris.ergoagri;

import android.app.Activity;

import android.content.Intent;

import android.graphics.Color;

import android.support.v7.app.AppCompatActivity;

import android.os.Bundle;

import android.view.Menu;

import android.view.MenuItem;

import android.widget.Toast;

import com.jjoe64.graphview.GraphView;

import com.jjoe64.graphview.series.DataPoint;

import com.jjoe64.graphview.series.DataPointInterface;

import com.jjoe64.graphview.series.LineGraphSeries;

import com.jjoe64.graphview.series.OnDataPointTapListener;

import com.jjoe64.graphview.series.Series;

public class HumidGraph extends Activity {

int[] date;

double[] humid;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_humid\_graph);

GraphView mygraph = (GraphView) findViewById(R.id.humidGraph);

Intent i = getIntent();

date=i.getIntArrayExtra("date");

humid=i.getDoubleArrayExtra("humid");

//create the array of datapoints to display data.

DataPoint[] mydata = new DataPoint[date.length];

for(int x=0;x<date.length;x++)

{

mydata[x]=new DataPoint(date[x],humid[x]);

}

//setting the parameters for the graph itself

LineGraphSeries<DataPoint> series = new LineGraphSeries<DataPoint>(mydata);

series.setColor(Color.CYAN);

mygraph.getViewport().setXAxisBoundsManual(true);

mygraph.getViewport().setMinX(0);

mygraph.getViewport().setMaxX(date.length);

mygraph.getViewport().setYAxisBoundsManual(true);

mygraph.getViewport().setMinY(0);

mygraph.getViewport().setMaxY(100);

mygraph.addSeries(series);

//displays a toast when the user presses on the graph to show the exact data at that point.

series.setOnDataPointTapListener(new OnDataPointTapListener() {

@Override

public void onTap(Series series, DataPointInterface dataPoint) {

Toast.makeText(HumidGraph.this, dataPoint.getY() + "%, " + (int) dataPoint.getX() + " hour(s) ago.", Toast.LENGTH\_SHORT).show();

}

});

}

@Override

public boolean onCreateOptionsMenu(Menu menu) {

// Inflate the menu; this adds items to the action bar if it is present.

getMenuInflater().inflate(R.menu.menu\_humid\_graph, menu);

return true;

}

@Override

public boolean onOptionsItemSelected(MenuItem item) {

// Handle action bar item clicks here. The action bar will

// automatically handle clicks on the Home/Up button, so long

// as you specify a parent activity in AndroidManifest.xml.

int id = item.getItemId();

//noinspection SimplifiableIfStatement

if (id == R.id.action\_settings) {

return true;

}

return super.onOptionsItemSelected(item);

}

}

### 7.7.5 Light History

package com.example.chris.ergoagri;

import android.app.Activity;

import android.content.Intent;

import android.content.pm.ActivityInfo;

import android.content.res.Configuration;

import android.support.v7.app.AppCompatActivity;

import android.os.Bundle;

import android.text.method.ScrollingMovementMethod;

import android.view.Menu;

import android.view.MenuItem;

import android.view.View;

import android.widget.TextView;

public class LightHistory extends Activity {

int[] date;

double[] light;

String display;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_light\_history);

if (tablet()) {

setRequestedOrientation(ActivityInfo.SCREEN\_ORIENTATION\_LANDSCAPE);

}

Intent i = getIntent();

date=i.getIntArrayExtra("date");

light=i.getDoubleArrayExtra("light");

TextView text = (TextView) findViewById(R.id.textViewLight2);

text.setMovementMethod(new ScrollingMovementMethod());

display = (date[0]) + "\t\t\t\t" + light[0] + "\n";

for(int x=1;x<date.length;x++)

{

display += (date[x])+ "\t\t\t\t" + light[x] + "\n";

}

text.setText(display);

}

//Checks to see if the device is a tablet and returns true if it is

private boolean tablet() {

return (this.getResources().getConfiguration().screenLayout

& Configuration.SCREENLAYOUT\_SIZE\_MASK)

>= Configuration.SCREENLAYOUT\_SIZE\_LARGE;

}

public void gotoLightGraph(View view)

{

Intent intent = new Intent(LightHistory.this, LightGraph.class);

intent.putExtra("date",date);

intent.putExtra("light",light);

LightHistory.this.startActivity(intent);

}

@Override

public boolean onCreateOptionsMenu(Menu menu) {

// Inflate the menu; this adds items to the action bar if it is present.

getMenuInflater().inflate(R.menu.menu\_light\_history, menu);

return true;

}

@Override

public boolean onOptionsItemSelected(MenuItem item) {

// Handle action bar item clicks here. The action bar will

// automatically handle clicks on the Home/Up button, so long

// as you specify a parent activity in AndroidManifest.xml.

int id = item.getItemId();

//noinspection SimplifiableIfStatement

if (id == R.id.action\_settings) {

return true;

}

return super.onOptionsItemSelected(item);

}

}

### 7.7.6 Light Graph

package com.example.chris.ergoagri;

import android.app.Activity;

import android.content.Intent;

import android.graphics.Color;

import android.support.v7.app.AppCompatActivity;

import android.os.Bundle;

import android.view.Menu;

import android.view.MenuItem;

import android.widget.Toast;

import com.jjoe64.graphview.GraphView;

import com.jjoe64.graphview.series.DataPoint;

import com.jjoe64.graphview.series.DataPointInterface;

import com.jjoe64.graphview.series.LineGraphSeries;

import com.jjoe64.graphview.series.OnDataPointTapListener;

import com.jjoe64.graphview.series.Series;

public class LightGraph extends Activity {

int[] date;

double[] light;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_light\_graph);

GraphView mygraph = (GraphView) findViewById(R.id.lightGraph);

Intent i = getIntent();

date=i.getIntArrayExtra("date");

light=i.getDoubleArrayExtra("light");

//create the array of datapoints to display data.

DataPoint[] mydata = new DataPoint[date.length];

for(int x=0;x<date.length;x++)

{

mydata[x]=new DataPoint(date[x],light[x]);

}

//setting the parameters for the graph itself

LineGraphSeries<DataPoint> series = new LineGraphSeries<DataPoint>(mydata);

series.setColor(Color.YELLOW);

mygraph.getViewport().setXAxisBoundsManual(true);

mygraph.getViewport().setMinX(0);

mygraph.getViewport().setMaxX(date.length);

mygraph.getViewport().setYAxisBoundsManual(true);

mygraph.getViewport().setMinY(0);

mygraph.getViewport().setMaxY(250);

mygraph.addSeries(series);

//displays a toast when the user presses on the graph to show the exact data at that point.

series.setOnDataPointTapListener(new OnDataPointTapListener() {

@Override

public void onTap(Series series, DataPointInterface dataPoint) {

Toast.makeText(LightGraph.this, "Light level: " + (int)dataPoint.getY() + ", " + (int) dataPoint.getX() + " hour(s) ago.", Toast.LENGTH\_SHORT).show();

}

});

}

@Override

public boolean onCreateOptionsMenu(Menu menu) {

// Inflate the menu; this adds items to the action bar if it is present.

getMenuInflater().inflate(R.menu.menu\_light\_graph, menu);

return true;

}

@Override

public boolean onOptionsItemSelected(MenuItem item) {

// Handle action bar item clicks here. The action bar will

// automatically handle clicks on the Home/Up button, so long

// as you specify a parent activity in AndroidManifest.xml.

int id = item.getItemId();

//noinspection SimplifiableIfStatement

if (id == R.id.action\_settings) {

return true;

}

return super.onOptionsItemSelected(item);

}

}

### 7.7.7 Soil Moisture History

package com.example.chris.ergoagri;

import android.app.Activity;

import android.content.Intent;

import android.content.pm.ActivityInfo;

import android.content.res.Configuration;

import android.support.v7.app.AppCompatActivity;

import android.os.Bundle;

import android.text.method.ScrollingMovementMethod;

import android.view.Menu;

import android.view.MenuItem;

import android.view.View;

import android.widget.TextView;

public class MoistureHistory extends Activity {

int[] date;

double[] moist;

String display;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_moisture\_history);

if (tablet()) {

setRequestedOrientation(ActivityInfo.SCREEN\_ORIENTATION\_LANDSCAPE);

}

Intent i = getIntent();

date=i.getIntArrayExtra("date");

moist=i.getDoubleArrayExtra("moist");

TextView text = (TextView) findViewById(R.id.textViewMoisture2);

text.setMovementMethod(new ScrollingMovementMethod());

display = (date[0]) + "\t\t\t\t" + moist[0] + "%" + "\n";

for(int x=1;x<date.length;x++)

{

display += (date[x])+ "\t\t\t\t" + moist[x] + "%" + "\n";

}

text.setText(display);

}

//Checks to see if the device is a tablet and returns true if it is

private boolean tablet() {

return (this.getResources().getConfiguration().screenLayout

& Configuration.SCREENLAYOUT\_SIZE\_MASK)

>= Configuration.SCREENLAYOUT\_SIZE\_LARGE;

}

public void gotoMoistureGraph(View view)

{

Intent intent = new Intent(MoistureHistory.this, MoistureGraph.class);

intent.putExtra("date",date);

intent.putExtra("moist",moist);

MoistureHistory.this.startActivity(intent);

}

@Override

public boolean onCreateOptionsMenu(Menu menu) {

// Inflate the menu; this adds items to the action bar if it is present.

getMenuInflater().inflate(R.menu.menu\_moisture\_history, menu);

return true;

}

@Override

public boolean onOptionsItemSelected(MenuItem item) {

// Handle action bar item clicks here. The action bar will

// automatically handle clicks on the Home/Up button, so long

// as you specify a parent activity in AndroidManifest.xml.

int id = item.getItemId();

//noinspection SimplifiableIfStatement

if (id == R.id.action\_settings) {

return true;

}

return super.onOptionsItemSelected(item);

}

}

### 7.7.8 Soil Moisture Graph

package com.example.chris.ergoagri;

import android.app.Activity;

import android.content.Intent;

import android.graphics.Color;

import android.support.v7.app.AppCompatActivity;

import android.os.Bundle;

import android.view.Menu;

import android.view.MenuItem;

import android.widget.Toast;

import com.jjoe64.graphview.GraphView;

import com.jjoe64.graphview.series.DataPoint;

import com.jjoe64.graphview.series.DataPointInterface;

import com.jjoe64.graphview.series.LineGraphSeries;

import com.jjoe64.graphview.series.OnDataPointTapListener;

import com.jjoe64.graphview.series.Series;

public class MoistureGraph extends Activity {

int[] date;

double[] moist;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_moisture\_graph);

GraphView mygraph = (GraphView) findViewById(R.id.moistGraph);

Intent i = getIntent();

date=i.getIntArrayExtra("date");

moist=i.getDoubleArrayExtra("moist");

//create the array of datapoints to display data.

DataPoint[] mydata = new DataPoint[date.length];

for(int x=0;x<date.length;x++)

{

mydata[x]=new DataPoint(date[x],moist[x]);

}

//setting the parameters for the graph itself

LineGraphSeries<DataPoint> series = new LineGraphSeries<DataPoint>(mydata);

series.setColor(Color.BLUE);

mygraph.getViewport().setXAxisBoundsManual(true);

mygraph.getViewport().setMinX(0);

mygraph.getViewport().setMaxX(date.length);

mygraph.getViewport().setYAxisBoundsManual(true);

mygraph.getViewport().setMinY(0);

mygraph.getViewport().setMaxY(100);

mygraph.addSeries(series);

//displays a toast when the user presses on the graph to show the exact data at that point.

series.setOnDataPointTapListener(new OnDataPointTapListener() {

@Override

public void onTap(Series series, DataPointInterface dataPoint) {

Toast.makeText(MoistureGraph.this, dataPoint.getY() + "%, " + (int) dataPoint.getX() + " hour(s) ago.", Toast.LENGTH\_SHORT).show();

}

});

}

@Override

public boolean onCreateOptionsMenu(Menu menu) {

// Inflate the menu; this adds items to the action bar if it is present.

getMenuInflater().inflate(R.menu.menu\_moisture\_graph, menu);

return true;

}

@Override

public boolean onOptionsItemSelected(MenuItem item) {

// Handle action bar item clicks here. The action bar will

// automatically handle clicks on the Home/Up button, so long

// as you specify a parent activity in AndroidManifest.xml.

int id = item.getItemId();

//noinspection SimplifiableIfStatement

if (id == R.id.action\_settings) {

return true;

}

return super.onOptionsItemSelected(item);

}

}

### 7.7.9 Temperature History

package com.example.chris.ergoagri;

import android.app.Activity;

import android.content.Intent;

import android.content.pm.ActivityInfo;

import android.content.res.Configuration;

import android.support.v7.app.AppCompatActivity;

import android.os.Bundle;

import android.text.method.ScrollingMovementMethod;

import android.view.Menu;

import android.view.MenuItem;

import android.view.View;

import android.widget.TextView;

public class TempHistory extends Activity {

int[] date;

double[] temp;

String display;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_temp\_history);

if (tablet()) {

setRequestedOrientation(ActivityInfo.SCREEN\_ORIENTATION\_LANDSCAPE);

}

Intent i = getIntent();

date=i.getIntArrayExtra("date");

temp=i.getDoubleArrayExtra("temp");

TextView text = (TextView) findViewById(R.id.textViewTemp2);

text.setMovementMethod(new ScrollingMovementMethod());

display = (date[0]) + "\t\t\t\t" + temp[0] + "°C" + "\n";

for(int x=1;x<date.length;x++)

{

display += (date[x])+ "\t\t\t\t" + temp[x] + "°C" + "\n";

}

text.setText(display);

}

public void gotoTempGraph(View view)

{

Intent intent = new Intent(TempHistory.this,TempGraph.class);

intent.putExtra("date", date);

intent.putExtra("temp",temp);

TempHistory.this.startActivity(intent);

}

//Checks to see if the device is a tablet and returns true if it is

private boolean tablet() {

return (this.getResources().getConfiguration().screenLayout

& Configuration.SCREENLAYOUT\_SIZE\_MASK)

>= Configuration.SCREENLAYOUT\_SIZE\_LARGE;

}

@Override

public boolean onCreateOptionsMenu(Menu menu) {

// Inflate the menu; this adds items to the action bar if it is present.

getMenuInflater().inflate(R.menu.menu\_temp\_history, menu);

return true;

}

@Override

public boolean onOptionsItemSelected(MenuItem item) {

// Handle action bar item clicks here. The action bar will

// automatically handle clicks on the Home/Up button, so long

// as you specify a parent activity in AndroidManifest.xml.

int id = item.getItemId();

//noinspection SimplifiableIfStatement

if (id == R.id.action\_settings) {

return true;

}

return super.onOptionsItemSelected(item);

}

}

### 7.7.10 Temperature Graph

package com.example.chris.ergoagri;

import android.app.Activity;

import android.content.Intent;

import android.graphics.Color;

import android.support.v7.app.AppCompatActivity;

import android.os.Bundle;

import android.view.Menu;

import android.view.MenuItem;

import android.widget.Toast;

import com.jjoe64.graphview.DefaultLabelFormatter;

import com.jjoe64.graphview.GraphView;

import com.jjoe64.graphview.series.DataPoint;

import com.jjoe64.graphview.series.DataPointInterface;

import com.jjoe64.graphview.series.LineGraphSeries;

import com.jjoe64.graphview.series.OnDataPointTapListener;

import com.jjoe64.graphview.series.Series;

public class TempGraph extends Activity {

int[] date;

double[] temp;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_temp\_graph);

GraphView mygraph = (GraphView) findViewById(R.id.tempGraph);

Intent i = getIntent();

date=i.getIntArrayExtra("date");

temp=i.getDoubleArrayExtra("temp");

//create the array of datapoints to display data.

DataPoint[] mydata = new DataPoint[date.length];

for(int x=0;x<date.length;x++)

{

mydata[x]=new DataPoint(date[x],temp[x]);

}

//setting the parameters for the graph itself

LineGraphSeries<DataPoint> series = new LineGraphSeries<DataPoint>(mydata);

series.setColor(Color.RED);

mygraph.getViewport().setXAxisBoundsManual(true);

mygraph.getViewport().setMinX(0);

mygraph.getViewport().setMaxX(date.length);

mygraph.getViewport().setYAxisBoundsManual(true);

mygraph.getViewport().setMinY(0);

mygraph.getViewport().setMaxY(40);

mygraph.addSeries(series);

//displays a toast when the user presses on the graph to show the exact data at that point.

series.setOnDataPointTapListener(new OnDataPointTapListener() {

@Override

public void onTap(Series series, DataPointInterface dataPoint) {

Toast.makeText(TempGraph.this, dataPoint.getY()+"°C, " + (int)dataPoint.getX() + " hour(s) ago.", Toast.LENGTH\_SHORT).show();

}

});

}

@Override

public boolean onCreateOptionsMenu(Menu menu) {

// Inflate the menu; this adds items to the action bar if it is present.

getMenuInflater().inflate(R.menu.menu\_temp\_graph, menu);

return true;

}

@Override

public boolean onOptionsItemSelected(MenuItem item) {

// Handle action bar item clicks here. The action bar will

// automatically handle clicks on the Home/Up button, so long

// as you specify a parent activity in AndroidManifest.xml.

int id = item.getItemId();

//noinspection SimplifiableIfStatement

if (id == R.id.action\_settings) {

return true;

}

return super.onOptionsItemSelected(item);

}

}

### 7.7.11 Style

<resources>

<!-- Base application theme. -->

<style name="AppTheme" parent="@android:style/Theme.Holo">

</style>

<style name="AppTheme.main">

<item name="android:layout\_width">wrap\_content</item>

<item name="android:layout\_height">wrap\_content</item>

<item name="android:textColor">@color/text</item>

<item name="android:background">@color/button</item>

<item name="android:padding">16dp</item>

<item name="android:typeface">serif</item>

<item name="android:textSize">14dp</item>

<item name="android:textStyle">bold</item>

</style>

</resources>

### 7.7.12 Text

<resources>

<string name="app\_name">ErgoAgri</string>

<string name="button1">Current Status</string>

<string name="button2">Temperature History</string>

<string name="button3">Light History</string>

<string name="button4">Moisture History</string>

<string name="button5">Humidity History</string>

<string name="button6">Update Data</string>

<string name="action\_settings">Settings</string>

<string name="title\_activity\_current\_data">Current Data</string>

<string name="hello\_world">Hello world!</string>

<string name="title\_activity\_light\_history">Light History</string>

<string name="title\_activity\_moisture\_history">Moisture History</string>

<string name="title\_activity\_humid\_history">Humid History</string>

<string name="title\_activity\_temp\_history">Temperature History</string>

<string name="Graph">Graph</string>

<string name="title\_activity\_light\_graph">Light Graph</string>

<string name="title\_activity\_temp\_graph">Temperature Graph</string>

<string name="title\_activity\_moisture\_graph">Moisture Graph</string>

<string name="title\_activity\_humid\_graph">Humid Graph</string>

</resources>

### 7.7.13 Vertical Main Menu Layout

<TableLayout xmlns:android="http://schemas.android.com/apk/res/android"

android:layout\_height="match\_parent" android:paddingLeft="@dimen/activity\_horizontal\_margin"

android:layout\_width="match\_parent"

android:paddingRight="@dimen/activity\_horizontal\_margin"

android:paddingTop="@dimen/activity\_vertical\_margin"

android:paddingBottom="@dimen/activity\_vertical\_margin"

android:background="@color/background">

<TableRow

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:padding="10dp"

android:gravity="center\_horizontal">

<Button

style="@style/AppTheme.main"

android:gravity="center\_horizontal"

android:text="@string/button1"

android:id="@+id/button1"

android:onClick="gotoCurrent"

>

</Button>

</TableRow>

<TableRow

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:padding="10dp"

android:gravity="center\_horizontal">

<Button

style="@style/AppTheme.main"

android:gravity="center\_horizontal"

android:text="@string/button2"

android:id="@+id/button2"

android:onClick="gotoTemp"

>

</Button>

</TableRow>

<TableRow

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:padding="10dp"

android:gravity="center\_horizontal">

<Button

style="@style/AppTheme.main"

android:gravity="center\_horizontal"

android:text="@string/button3"

android:id="@+id/button3"

android:onClick="gotoLight"

>

</Button>

</TableRow>

<TableRow

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:padding="10dp"

android:gravity="center\_horizontal">

<Button

style="@style/AppTheme.main"

android:gravity="center\_horizontal"

android:text="@string/button4"

android:id="@+id/button4"

android:onClick="gotoMoisture"

>

</Button>

</TableRow>

<TableRow

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:padding="10dp"

android:gravity="center\_horizontal">

<Button

style="@style/AppTheme.main"

android:gravity="center\_horizontal"

android:text="@string/button5"

android:id="@+id/button5"

android:onClick="gotoHumid"

>

</Button>

</TableRow>

<TableRow

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:padding="10dp"

android:gravity="center\_horizontal">

<Button

style="@style/AppTheme.main"

android:gravity="center\_horizontal"

android:text="@string/button6"

android:id="@+id/button6"

android:onClick="update"

>

</Button>

</TableRow>

</TableLayout>

### 7.7.14 Horizontal Main Menu Layout

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"

android:layout\_height="match\_parent" android:paddingLeft="@dimen/activity\_horizontal\_margin"

android:layout\_width="match\_parent"

android:paddingRight="@dimen/activity\_horizontal\_margin"

android:paddingTop="@dimen/activity\_vertical\_margin"

android:paddingBottom="@dimen/activity\_vertical\_margin"

android:background="@color/background"

android:orientation="vertical">

<RelativeLayout

android:layout\_width="fill\_parent"

android:layout\_height="wrap\_content"

android:padding="18dp">

<!-- In here will go the logo-->

</RelativeLayout>

<RelativeLayout

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:padding="15dp">

<Button

style="@style/AppTheme.main"

android:gravity="center\_horizontal"

android:text="@string/button1"

android:id="@+id/button1"

android:onClick="gotoCurrent"

android:layout\_marginEnd="40dp"

android:layout\_alignParentTop="true"

android:layout\_toStartOf="@+id/button6">

</Button>

<Button

style="@style/AppTheme.main"

android:text="@string/button6"

android:id="@+id/button6"

android:layout\_marginEnd="113dp"

android:layout\_alignParentTop="true"

android:layout\_alignParentEnd="true"

android:onClick="update"

>

</Button>

</RelativeLayout>

<RelativeLayout

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:padding="15dp">

<Button

style="@style/AppTheme.main"

android:gravity="center\_horizontal"

android:text="@string/button3"

android:id="@+id/button3"

android:onClick="gotoLight"

android:layout\_column="0"

android:layout\_marginEnd="40dp"

android:layout\_alignParentTop="true"

android:layout\_toStartOf="@+id/button4">

</Button>

<Button

style="@style/AppTheme.main"

android:gravity="center\_horizontal"

android:text="@string/button4"

android:id="@+id/button4"

android:onClick="gotoMoisture"

android:layout\_column="0"

android:layout\_marginEnd="79dp"

android:layout\_alignTop="@+id/button3"

android:layout\_alignParentEnd="true">

</Button>

</RelativeLayout>

<RelativeLayout

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:padding="15dp">

<Button

style="@style/AppTheme.main"

android:text="@string/button5"

android:id="@+id/button5"

android:onClick="gotoHumid"

android:paddingRight="25dp"

android:layout\_width="wrap\_content"

android:paddingLeft="25dp"

android:layout\_marginEnd="40dp"

android:layout\_alignParentTop="true"

android:layout\_toStartOf="@+id/button2">

</Button>

<Button

style="@style/AppTheme.main"

android:text="@string/button2"

android:id="@+id/button2"

android:onClick="gotoTemp"

android:paddingRight="25dp"

android:layout\_width="wrap\_content"

android:paddingLeft="25dp"

android:layout\_alignParentTop="true"

android:layout\_alignParentEnd="true"

android:layout\_marginEnd="50dp">

</Button>

</RelativeLayout>

</LinearLayout>

### 7.7.15 Vertical Current Data Layout

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:tools="http://schemas.android.com/tools" android:layout\_width="match\_parent"

android:layout\_height="match\_parent" android:paddingLeft="@dimen/activity\_horizontal\_margin"

android:paddingRight="@dimen/activity\_horizontal\_margin"

android:paddingTop="@dimen/activity\_vertical\_margin"

android:paddingBottom="@dimen/activity\_vertical\_margin"

tools:context="com.example.chris.ergoagri.CurrentData"

>

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="@string/button1"

android:id="@+id/textView"

android:layout\_gravity="center\_horizontal"

android:layout\_alignParentTop="true"

android:layout\_centerHorizontal="true"

android:textSize="24sp"

/>

<ImageView

android:layout\_width="100dp"

android:layout\_height="100dp"

android:id="@+id/imageView"

android:src="@drawable/temp"

android:layout\_above="@+id/imageView2"

android:layout\_alignParentStart="true" />

<ImageView

android:layout\_width="100dp"

android:layout\_height="100dp"

android:id="@+id/imageView2"

android:src="@drawable/humidity"

android:layout\_above="@+id/imageView3"

android:layout\_alignParentStart="true" />

<ImageView

android:layout\_width="100dp"

android:layout\_height="100dp"

android:id="@+id/imageView3"

android:src="@drawable/light"

android:layout\_above="@+id/imageView4"

android:layout\_alignParentStart="true" />

<ImageView

android:layout\_width="100dp"

android:layout\_height="100dp"

android:id="@+id/imageView4"

android:src="@drawable/moisture"

android:layout\_alignParentBottom="true"

android:layout\_alignParentStart="true" />

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="New Text"

android:id="@+id/curTemp"

android:layout\_alignTop="@+id/imageView"

android:layout\_above="@+id/imageView2"

android:layout\_alignParentEnd="true"

android:layout\_toEndOf="@+id/imageView"

android:gravity="center\_vertical"

android:textSize="24sp"

/>

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="New Text"

android:id="@+id/curHumid"

android:layout\_below="@+id/imageView"

android:layout\_alignBottom="@+id/imageView2"

android:layout\_alignParentEnd="true"

android:layout\_toEndOf="@+id/imageView"

android:gravity="center\_vertical"

android:textSize="24sp"

/>

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="New Text"

android:id="@+id/curLight"

android:layout\_alignTop="@+id/imageView3"

android:layout\_above="@+id/imageView4"

android:layout\_alignEnd="@+id/curMoist"

android:layout\_toEndOf="@+id/imageView3"

android:gravity="center\_vertical"

android:textSize="24sp"

/>

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="New Text"

android:id="@+id/curMoist"

android:layout\_below="@+id/imageView3"

android:layout\_alignBottom="@+id/imageView4"

android:layout\_alignParentEnd="true"

android:layout\_toEndOf="@+id/imageView3"

android:gravity="center\_vertical"

android:textSize="24sp"

/>

</RelativeLayout>

### 7.7.16 Horizontal Current Data Layout

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:tools="http://schemas.android.com/tools" android:layout\_width="match\_parent"

android:layout\_height="match\_parent" android:paddingLeft="@dimen/activity\_horizontal\_margin"

android:paddingRight="@dimen/activity\_horizontal\_margin"

android:paddingTop="@dimen/activity\_vertical\_margin"

android:paddingBottom="@dimen/activity\_vertical\_margin"

tools:context=".CurrentData"

android:orientation="vertical"

>

<RelativeLayout

android:layout\_width="fill\_parent"

android:layout\_height="fill\_parent"

android:layout\_weight="80">

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="@string/button1"

android:id="@+id/textView"

android:layout\_gravity="center\_horizontal"

android:textSize="24sp"

android:layout\_alignParentEnd="false"

android:layout\_alignParentStart="false"

android:layout\_centerHorizontal="true" />

</RelativeLayout>

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:orientation="horizontal"

android:layout\_weight="20"

android:layout\_gravity="bottom">

<LinearLayout

android:layout\_width="0dp"

android:layout\_height="match\_parent"

android:orientation="vertical"

android:layout\_weight=".5"

android:layout\_gravity="left">

<LinearLayout

android:layout\_width="wrap\_content"

android:layout\_height="fill\_parent"

android:layout\_weight="50"

android:orientation="horizontal"

android:weightSum="5"

android:layout\_gravity="top">

<ImageView

android:layout\_width="0dp"

android:layout\_height="match\_parent"

android:id="@+id/imageView"

android:src="@drawable/temp"

android:baselineAlignBottom="true"

android:layout\_weight="3"

android:adjustViewBounds="true"/>

<TextView

android:layout\_width="0dp"

android:layout\_height="wrap\_content"

android:text="New Text"

android:id="@+id/curTemp"

android:gravity="center\_vertical"

android:textSize="24sp"

android:layout\_weight="2"

android:layout\_gravity="center\_vertical" />

</LinearLayout>

<LinearLayout

android:layout\_width="wrap\_content"

android:layout\_height="fill\_parent"

android:layout\_weight="50"

android:orientation="horizontal">

<ImageView

android:layout\_width="0dp"

android:layout\_height="match\_parent"

android:id="@+id/imageView2"

android:src="@drawable/humidity"

android:baselineAlignBottom="true"

android:layout\_weight="3"

android:adjustViewBounds="true"/>

<TextView

android:layout\_width="0dp"

android:layout\_height="wrap\_content"

android:text="New Text"

android:id="@+id/curHumid"

android:gravity="center\_vertical"

android:textSize="24sp"

android:layout\_weight="2"

android:layout\_gravity="center\_vertical" />

</LinearLayout>

</LinearLayout>

<LinearLayout

android:layout\_width="0dp"

android:layout\_height="match\_parent"

android:orientation="vertical"

android:layout\_weight=".5"

android:layout\_gravity="right">

<LinearLayout

android:layout\_width="wrap\_content"

android:layout\_height="fill\_parent"

android:layout\_weight="50"

android:orientation="horizontal">

<ImageView

android:layout\_width="0dp"

android:layout\_height="match\_parent"

android:id="@+id/imageView3"

android:src="@drawable/light"

android:baselineAlignBottom="true"

android:layout\_weight="3"

android:adjustViewBounds="true"/>

<TextView

android:layout\_width="0dp"

android:layout\_height="wrap\_content"

android:text="New Text"

android:id="@+id/curLight"

android:gravity="center\_vertical"

android:textSize="24sp"

android:layout\_weight="2"

android:layout\_gravity="center\_vertical" />

</LinearLayout>

<LinearLayout

android:layout\_width="wrap\_content"

android:layout\_height="fill\_parent"

android:layout\_weight="50"

android:orientation="horizontal">

<ImageView

android:layout\_width="0dp"

android:layout\_height="match\_parent"

android:id="@+id/imageView4"

android:src="@drawable/moisture"

android:baselineAlignBottom="true"

android:layout\_weight="3"

android:adjustViewBounds="true"/>

<TextView

android:layout\_width="0dp"

android:layout\_height="wrap\_content"

android:text="New Text"

android:id="@+id/curMoist"

android:gravity="center\_vertical"

android:textSize="24sp"

android:layout\_weight="2"

android:layout\_gravity="center\_vertical" />

</LinearLayout>

</LinearLayout>

</LinearLayout>

</LinearLayout>

### 7.7.17 Humidity History Layout

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:tools="http://schemas.android.com/tools" android:layout\_width="match\_parent"

android:layout\_height="match\_parent" android:paddingLeft="@dimen/activity\_horizontal\_margin"

android:paddingRight="@dimen/activity\_horizontal\_margin"

android:paddingTop="@dimen/activity\_vertical\_margin"

android:paddingBottom="@dimen/activity\_vertical\_margin"

tools:context="com.example.chris.ergoagri.HumidHistory">

<TextView android:text="@string/button5"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_alignParentTop="true"

android:layout\_centerHorizontal="true"

android:id="@+id/textViewHumid" />

<Button

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="@string/Graph"

android:id="@+id/buttonHumid"

android:layout\_below="@+id/textViewHumid"

android:layout\_centerHorizontal="true"

android:onClick="gotoHumidGraph"

/>

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:id="@+id/textViewHumid2"

android:layout\_below="@+id/buttonHumid"

android:layout\_alignParentBottom="true"

android:layout\_alignParentEnd="true"

android:layout\_alignParentStart="true"

/>

</RelativeLayout>

### 7.7.18 Light History Layout

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:tools="http://schemas.android.com/tools" android:layout\_width="match\_parent"

android:layout\_height="match\_parent" android:paddingLeft="@dimen/activity\_horizontal\_margin"

android:paddingRight="@dimen/activity\_horizontal\_margin"

android:paddingTop="@dimen/activity\_vertical\_margin"

android:paddingBottom="@dimen/activity\_vertical\_margin"

tools:context="com.example.chris.ergoagri.LightHistory">

<TextView android:text="@string/button3"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_alignParentTop="true"

android:layout\_centerHorizontal="true"

android:id="@+id/textViewLight" />

<Button

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="@string/Graph"

android:id="@+id/buttonLight"

android:layout\_below="@+id/textViewLight"

android:layout\_centerHorizontal="true"

android:onClick="gotoLightGraph"

/>

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:id="@+id/textViewLight2"

android:layout\_below="@+id/buttonLight"

android:layout\_alignParentBottom="true"

android:layout\_alignParentEnd="true"

android:layout\_alignParentStart="true"

android:text="Oct 26: \t\t LOW \nOct 27 \t\t HIGH \nOct 28 \t\t HIGH \nOct 29 \t\t MEDIUM \nOct 30 \t\t LOW \nOct 31 \t\t HIGH \nNov 1 \t\t HIGH \nNov 2 \t\t MEDIUM"

/>

</RelativeLayout>

### 7.7.19 Moisture History Layout

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:tools="http://schemas.android.com/tools" android:layout\_width="match\_parent"

android:layout\_height="match\_parent" android:paddingLeft="@dimen/activity\_horizontal\_margin"

android:paddingRight="@dimen/activity\_horizontal\_margin"

android:paddingTop="@dimen/activity\_vertical\_margin"

android:paddingBottom="@dimen/activity\_vertical\_margin"

tools:context="com.example.chris.ergoagri.MoistureHistory">

<TextView android:text="@string/button4"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_alignParentTop="true"

android:layout\_centerHorizontal="true"

android:id="@+id/textViewMoisture" />

<Button

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="@string/Graph"

android:id="@+id/buttonMoisture"

android:layout\_below="@+id/textViewMoisture"

android:layout\_centerHorizontal="true"

android:onClick="gotoMoistureGraph"

/>

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:id="@+id/textViewMoisture2"

android:layout\_below="@+id/buttonMoisture"

android:layout\_alignParentBottom="true"

android:layout\_alignParentEnd="true"

android:layout\_alignParentStart="true"

android:text="Oct 26: \t\t 70% \nOct 27 \t\t 74% \nOct 28 \t\t 68% \nOct 29 \t\t 90% \nOct 30 \t\t 57% \nOct 31 \t\t 66% \nNov 1 \t\t 40% \nNov 2 \t\t 77%"

/>

</RelativeLayout>

### 7.7.20 Temperature History Layout

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:tools="http://schemas.android.com/tools" android:layout\_width="match\_parent"

android:layout\_height="match\_parent" android:paddingLeft="@dimen/activity\_horizontal\_margin"

android:paddingRight="@dimen/activity\_horizontal\_margin"

android:paddingTop="@dimen/activity\_vertical\_margin"

android:paddingBottom="@dimen/activity\_vertical\_margin"

tools:context="com.example.chris.ergoagri.TempHistory">

<TextView android:text="@string/button2"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_alignParentTop="true"

android:layout\_centerHorizontal="true"

android:id="@+id/textViewTemp" />

<Button

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="@string/Graph"

android:id="@+id/buttonTemp"

android:layout\_below="@+id/textViewTemp"

android:layout\_centerHorizontal="true"

android:onClick="gotoTempGraph"

/>

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:id="@+id/textViewTemp2"

android:layout\_below="@+id/buttonTemp"

android:layout\_alignParentBottom="true"

android:layout\_alignParentEnd="true"

android:layout\_alignParentStart="true"

android:text="Oct 26: \t\t 21C \nOct 27 \t\t 24C \nOct 28 \t\t 22C \nOct 29 \t\t 19C \nOct 30 \t\t 16C \nOct 31 \t\t 14C \nNov 1 \t\t 14C \nNov 2 \t\t 11C"

/>

</RelativeLayout>

### 7.7.21 Humidity Graph Layout

<?xml version="1.0" encoding="utf-8"?>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:app="http://schemas.android.com/apk/res-auto"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent">

<com.jjoe64.graphview.GraphView

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:id="@+id/humidGraph"

/>

</RelativeLayout>

### 7.7.22 Light Graph Layout

<?xml version="1.0" encoding="utf-8"?>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:app="http://schemas.android.com/apk/res-auto"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent">

<com.jjoe64.graphview.GraphView

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:id="@+id/lightGraph"

/>

</RelativeLayout>

### 7.7.23 Moisture Graph Layout

<?xml version="1.0" encoding="utf-8"?>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:app="http://schemas.android.com/apk/res-auto"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent">

<com.jjoe64.graphview.GraphView

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:id="@+id/moistGraph"

/>

</RelativeLayout>

### 7.7.24 Temperature Graph Layout

<?xml version="1.0" encoding="utf-8"?>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:app="http://schemas.android.com/apk/res-auto"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent">

<com.jjoe64.graphview.GraphView

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:id="@+id/tempGraph"

/>

</RelativeLayout>